ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Creación de una aventura gráfica integrando vídeo real

Proyecto final de carrera

Ingeniería técnica de telecomunicación, especialidad en sonido e imagen

Pamplona, 9 de Septiembre de 2011

Dr. Mikel Sagüés García

Irene Perea Castellanos

ÍNDICE

Capítulo 1: DESCRIPCIÓN DEL PROYECTO

Este proyecto consiste en la creación de un juego, en concreto de una aventura gráfica, en el que se mezcla animación con video real. Los escenarios y objetos que aparecen en el juego están dibujados, sin embargo el protagonista y los personajes son vídeos reales que han sido grabados en el plató de televisión de la universidad y posteriormente editados para poder incluirlos e interactuar con ellos en la aventura gráfica.

En el **capítulo 2**, se hace una descripción de este tipo de juegos, las aventuras gráficas y su evolución a lo largo de los años. También hay referencias a aventuras gráficas realizadas en Flash y aplicaciones que incluyen vídeo real. Para esta parte se consultaron numerosas páginas web y tutoriales de Flash y ActionScript, sobre la creación de juegos, puzzles personajes y animaciones.

En el **capítulo 3** se habla de las tecnologías que se han utilizado para la creación de la aventura gráfica, estas son: el lenguaje de programación ActionScript 3.0 y los programas Adobe Flash CS5 y Adobe Premiere CS5.

Toda la aventura está creada sobre la plataforma Adobe Flash CS5, los escenarios, objetos y determinados personajes, han sido creados mediante las herramientas de dibujo que ofrece Flash, así como las animaciones que aparecen al principio y al final del juego. Esta plataforma utiliza el lenguaje de programación orientado a objetos ActionScript 3.0 y con el se han creado todas las acciones de los objetos, personajes y la interactividad entre ambos.

Para incluir el vídeo real sobre un fondo dibujado en Flash, se ha usado la técnica conocida como Chroma key, que permite grabar a una persona sobre un fondo y posteriormente en la fase de edición, eliminar ese fondo y sustituirlo por otro. En el **capítulo 4** se hace una definición detallada y se explican los aspecto más importantes para su correcta realización.

Para la edición de los vídeos y realizar la técnica del Chroma key se ha usado el editor Adobe Premiere CS5.

En el **capítulo 5,** se describen una serie de ensayos, previos al diseño de la aventura gráfica, que abarcan todo lo necesario para su desarrollo final. Las pruebas están enfocadas al aprendizaje del lenguaje ActionScript 3.0, a la manipulación de vídeo en Flash, manejo de la cámara y edición de vídeos. Esta fase fue a la que más tiempo se le dedicó, ya que era muy importante comprender los aspectos descrito, ya que de ellos, dependería el resultado final de la aventura.

Una vez se realizaron y se analizaron todas las pruebas, se comenzó a diseñar la aventura gráfica. Todo este proceso viene detallado en el **capítulo 6**. En este punto se decidió la historia que se quería contar, los escenarios en los que se iba a suceder, que objetos y personajes, reales y animados iban a aparecer, así como las formas en las que se podría interactuar con los objetos y personajes. Para todo esto se realizó un guión artístico y otro técnico.

La última fase, fue la del desarrollo propio de la aventura gráfica en función de los resultados de las pruebas y el diseño realizado. El **capítulo 7** muestra la estructura en base a la programación que posee el juego. También se describen las funciones más importantes y la lógica seguida para hacer que los objetos mantengan su estado o lo cambien en función de las acciones realizadas o el escenario en el que se encuentre el personaje.

Por último, se incluye en el **capítulo 8** una serie de conclusiones sacadas tras la realización del proyecto. Estas conclusiones engloban temas generales sobre el trabajo realizado, temas más técnicos referidos a las pruebas realizadas y a la parte de la programación y para terminar, conclusiones personales sobre el proyecto en general.

Capítulo 2: INTRODUCCIÓN A LAS AVENTURAS GRÁFICAS

Este proyecto se basa en la realización de un juego, en particular, una aventura gráfica. La dinámica de este tipo de juego consiste en ir avanzando a medida que se van resolviendo puzzles, rompecabezas y diferentes retos que se suceden en la historia, interactuando con personajes y objetos a través de un menú de acciones, como por ejemplo, *usar, hablar, coger,* utilizando el teclado o el ratón para mover el personaje.

Breve historia de las aventuras gráficas

Las primeras aventuras gráficas se remontan a los años setenta. El sistema era sencillo, constaba de diálogos y acciones que funcionaban a través de comandos, como "abrir puerta", "hablar con", "coger", etc. Un buen ejemplo es la saga **Zork.**

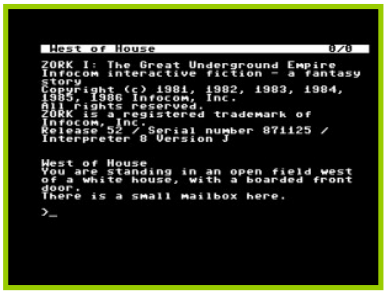


Figura 2.1. Zork

Los videojuegos con imágenes no llegarían hasta 1980. Tampoco aportaban mucho más, sólo gráficos. Lo más importante es que supusieron el nacimiento de un nuevo género. Una aventura muy popular de esta tipo es **Mystery House**.

Figura 2.2. Mistery House

La revolución llegó en 1987 con la creación de **Lucasfilm Games** (la actual LucasArts). La empresa creó un motor diferente el llamado **SCUMM, [A01]** (Script Utility for Maniac Mansion) y el **point&click [A02]** que supuso el gran salto de la aventura de texto a la de ratón. Los jugadores ya podían interactuar directamente con los gráficos a base de "clicks".

Una de las a venturas más famosas creadas por la empresa Lucasart es: **The Secret of Monkey Island**, el juego caracterizado por su humor un tanto absurdo, sus personajes emblemáticos, o las geniales peleas de insultos. La primera entrega de la serie apareció en 1990, y en 2009 la compañía lo remasterizó con un resultado muy bueno.

Figura 2.3. Monkey Island

LucasArts también fueron los creadores de grandes clásicos de la aventura gráfica como **Sam & Max**, **The Day of Tentacle**, **Maniac Mansion** o la saga de **Indiana Jones**. En 1998 la compañía abandonó los gráficos planos y creo su primera a ventura en 3D, **Grim Fandango.**

En esta época las 2D eran la manera estándar de visualización del género, cosa que comenzó a cambiar en 1992 cuando **Infogrames** lanzó **Alone in the Dark**, juego que va más allá de las aventuras gráficas, en el que hay que resolver enigmas pero también librarse de monstruos, zombies, etc.. Se trata de un juego revolucionario que permite por primera vez en una aventura jugar con diferentes ángulos de cámara.

Figura 2.4. Alone in the Dark

Las nuevas propuestas comienzan a llegar de la mano de **Access Software** que en 1994 saca **Under a Killing Moon,** una aventura gráfica con una jugabilidad clásica y una perspectiva en 1ª persona, gráficos en 3D y tiempo real, un motor flexible que permitía al jugador moverse en cualquier dirección, **Full-Motion Video (FMV),** y voces para todos los diálogos. aunque mantenía la herencia clásica: los puzzles están basados en los diálogos y el uso del inventario y la trama sigue siendo parte fundamental del juego. Access Software utilizó el mismo motor para otras dos grandes aventura FMV, **The Pandora Directive** en 1996 **y Overseer** en 1998.

The 7th Guest, fue un juego revolucionario que intentó cambiar el género completamente. Equipado con lo último en tecnología infográfica, este juego se basaba no en el diálogo ni en el uso de objetos sino en la resolución de puzzles independientes cuya solución requería combinar, manipular o ajustar piezas, mover palancas, etc. El diálogo se redujo al mínimo, y en su lugar este tipo de puzle se convirtió en el centro del juego. La atmósfera y la resolución de puzzles tenían más importancia que la historia o la interacción con el entorno.

Figura 2.5. The 7th Guest

Otro mito dentro de este género es **Myst** que ofrece visión en primera persona y la navegación con cursores hacia nuevas imágenes (slideshow), pantallas o diapositivas parcas en puntos interactivos (hotspots), mucha exploración y observación de las pistas disimuladas en el entorno, y resolución de puzzles de maquinaria.

Figura 2.6. Myst

Como se ve, las aventuras han evolucionado de una forma sorprendente y hay infinidad de modelos y de tipos, ya que muchas combinan los elementos tradicionales con nuevas técnicas, por lo que las opciones son muy variadas.

La aventura que se pretende diseñar, estará inspirada en las aventuras creadas por *Lucasart* (*The Secret Of Monkey Island, Day Of The Tentacle, Sam & Max Hit the Road…)* por su originalidad, humor, dificultad e ingenio en la resolución de puzzles.

Para realizar una, se tendrá que pensar antes en cosas muy concretas, como: la historia que se quiere contar, es decir, se tendrá que realizar un guión artístico y técnico, como va a ser la interfaz de la aventura gráfica, las acciones que podrá hacer el personaje, los puzzles, y lo más importante como va a programarse todo esto, con que y si es posible, para esta última parte se deberá hacer a priori una serie de pruebas básicas, para cerciorarse de que se puede hacer lo que se ha pensado. Existen páginas en Internet donde dan información e ideas sobre los pasos a seguir, **[A04-A05]**, tipos de puzzles que se pueden diseñar **[A06]**, la creación de un guión para videojuego, etc. **[A07].**

Teniendo en cuenta lo anterior, lo primero que se hizo fue buscar aventuras gráficas realizadas en flash, para ver que existe en la Web y en el mercado y que puede llegar a conseguirse. Algunas de las que se encontraron aparecen en **[A9-A15]**. También existen páginas que recopilan muchos juegos en flash **[A16-A18].**

Mirando este tipo de páginas se encontraron dos blogs de dos personas que estaban realizando una aventura gráfica, y en ellos explican los pasos que han ido dando, las dificultades que han tenido y la evolución de su trabajo [A19-A20]. Es interesante, para tener en cuenta lo que hay que hacer, saber organizar cada parte y tener en cuenta los problemas que puedan surgir.

Un aspecto muy importante es elegir el motor con el que se va a crear la aventura. Existen mucho programas específicos para crear aventuras gráficas en los que no es necesario programar como: Wintermute Engine Devolpment Kit.AGS, Adventure Game Studio, Game Maker, 3DGameStudio. Kodu y muchos más [A8]. Todos estos motores son fáciles de usar y pueden dar resultados muy atractivos pero, tendrán muchas limitaciones, aunque alguno de ellos como el Game Maker permite desarrollar acciones mediante el lenguaje propio Game Maker Language.

La innovación que se quiere introducir en nuestra aventura gráfica es mezclar video real con animación, para ello se estudiarán las herramientas de Flash que permitan la manipulación de vídeo y que más tarde se detallarán.

Existen en la Web multitud de aplicaciones interactivas que mezclan animación e imagen real, algunas de ellas aparecen en los siguientes enlaces [A21 a A26].

Para la parte de programación del proyecto primero se consultó un libro, donde te adentraba en los aspectos más importantes de ActionScript 3, explicando las bases y con ejemplos prácticos [F1], también páginas similares, con tutoriales variados de ActionScript 3 [F11-F16], además de los siguientes manuales [F17-F19]. Más tarde tutoriales más concretos para realizar las pruebas necesarias que se creyeron indispensables para la aventura, tutoriales sobre: profundidad en flash, creación de botones, interactividad con el ratón, crear un espacio de dibujo, trabajar con texto, sonidos e interpolaciones [F20-F43].

Para la parte artística, se consultaron tutoriales sobre animación en flash, creación de personajes y efectos sobre ellos [85 a 100] y también animación con imágenes reales [101 a 102].

Capítulo 3: INTRODUCCIÓN A LAS TECNOLOGÍAS

Para llevar a cabo el desarrollo de la aventura gráfica se necesitaba un entorno en el que crearla y un lenguaje de programación que permitiera manipular objetos y sus propiedades. Por eso se eligió el entorno de **Adobe Flash CS5** y el lenguaje de programación **ActionScript 3.0**. También se necesitaba un editor de vídeo potente, capaz de exportar vídeos en el formato adecuado para ser reproducidos en Flash, el programa elegido fue **Adobe Premiere Pro CS5.**

Adobe Flash CS5

Programa de edición multimedia que permite la creación de aplicaciones interactivas, que utiliza principalmente gráficos vectoriales, pero también imágenes ráster, sonido, código de programa, flujo de vídeo y audio bidireccional para crear proyectos multimedia. Flash es el entorno desarrollador y Flash Player es el programa (la máquina virtual) utilizado para ejecutar los archivos generados con Flash.

Los proyectos multimedia pueden ser desde simples animaciones hasta complejos programas pues, además de los gráficos, vídeos y sonidos, Flash incorpora ActionScript, un completo lenguaje de programación que amplía enormemente las posibilidades en los proyectos.

Los archivos de Flash suelen tener la extensión SWF y aparecen frecuentemente en páginas web en forma de animaciones y aplicaciones

Sabiendo todo lo que ofrece Flash, se creará la aventura gráfica, con el, ya que aporta todo lo que se necesita. Un entorno para crear aplicaciones interactivas, en el que se puede incluir y manipular vídeo real y que ofrece un potente lenguaje de programación para diseñar toda la interactividad entre los diferentes elementos que se creen.

Los escenarios, objetos y determinados personajes se diseñarán con las herramientas de dibujo que ofrezca Flash.

Como se ha comentado, para programar la aventura gráfica se usará el lenguaje ActionScrip. Adobe Flash CS5, ofrece la versión ActionScript 3.0.

ActionScript 3.0

Es un lenguaje de programación orientado a objetos **[F2-F3].** Este lenguaje, que ha sido mejorado respecto a su versión anterior, ActionScript 2.0. Se basa en el uso de clases **[F6-F9]** y tiene posibilidades casi infinitas para la realización de este tipo de juegos.

3.1. PROGRAMACIÓN ORIENTADA A OBJETOS.

La programación Orientada a objetos (POO) es una forma especial de programar, más cercana a como se expresarían las cosas en la vida real que otros tipos de programación.

La manera de programar se basa en términos de objetos, propiedades y métodos principalmente.

Para que se entienda mejor se pondrá un ejemplo de un coche para tratar de modelizarlo en un esquema de POO. El coche es el elemento principal que tiene una serie de

características, como podrían ser el color, el modelo o la marca. Además tiene una serie de funcionalidades asociadas, como pueden ser ponerse en marcha, parar o aparcar.

En un esquema POO el coche sería el objeto, las propiedades serían las características como el color o el modelo y los métodos serían las funcionalidades asociadas como ponerse en marcha o parar.

3.1.1. Clases:

Las clases son declaraciones de objetos. Esto quiere decir que la definición de un objeto es la clase. Cuando se programa un objeto y se definen sus características y funcionalidades en realidad lo que se está haciendo es programar una clase.

3.1.2. Propiedades:

Las propiedades o atributos son las características de los objetos. Cuando se define una propiedad normalmente se especifica su nombre y su tipo. Las propiedades son variables donde se almacenan datos relacionados con los objetos.

3.1.3. Métodos:

Los métodos son las funciones que están asociadas a un objeto.

3.1.4. Objetos:

Los objetos son ejemplares de una clase cualquiera. Cuando se crea un ejemplar se tiene que especificar la clase a partir de la cual se creará. Esta acción de crear un objeto a partir de una clase se llama instanciar.

Para crear un objeto se tiene que escribir una instrucción especial, en ActionScript 3.0, se haría de la siguiente forma:

```
miCoche = new Coche()
```

Con la palabra new se especifica que se tiene que crear una instancia de la clase que sigue a continuación. Dentro de los paréntesis se pueden colocar parámetros con los que inicializar el objeto de la clase coche.

3.1.5. Estados en objetos

Cuando se tiene un objeto, sus propiedades toman valores. Por ejemplo, con el objeto coche, la propiedad color tomará un valor en concreto, como por ejemplo rojo. El valor concreto de una propiedad de un objeto se llama estado.

Para acceder a un estado de un objeto para ver su valor o cambiarlo se utiliza el operador punto.

```
miCoche.color = rojo
```

El objeto es miCoche, luego se coloca el operador punto y por último el nombre de la propiedad a la que se desea acceder.

3.1.6. Mensajes en objetos

Un mensaje en un objeto es la acción de efectuar una llamada a un método. Por ejemplo, cuando se le dice a un objeto coche que se ponga en marcha se está pasando el mensaje "ponte en marcha".

Para mandar mensajes a los objetos se utiliza el operador punto, seguido del método que se desea invocar.

```
miCoche.ponerseEnMarcha()
```

En este ejemplo se pasa el mensaje ponerseEnMarcha(). Hay que colocar paréntesis igual que cualquier llamada a una función, dentro irían los parámetros.

3.1.7. Otras características

La **herencia** sirve para crear objetos que incorporen propiedades y métodos de otros objetos. Así se puede construir unos objetos a partir de otros sin tener que reescribirlo todo.

El **polimorfismo** es la capacidad de referirse a objetos de clases distintas en una jerarquía utilizando el mismo elemento de programa para realizar la misma operación, pero de formas distintas. Esto sirve para que no preocuparse sobre lo que se está trabajando, y definir un código que sea compatible con objetos de varios tipos.

3.2. CONTROL DE VÍDEO

Además de todo esto, Flash permite la utilización y la manipulación de video a través de Actionscript 3.0 de varias formas. Una de ellas, y la que se utilizará es a través del componente FLVPlayback [V1-V2]:

3.2.1. Componente FLVPlayback,

Permite incluir fácilmente un reproductor de vídeo en la aplicación Flash para reproducir archivos de vídeo Flash (FLV). Tiene las siguientes características y ventajas:

- Puede arrastrarse al escenario e implementarse rápidamente de forma correcta.
- Proporciona una colección de *aspectos* prediseñados que permiten personalizar la apariencia de los controles de reproducción.
- Permite a los usuarios avanzados crear sus propios aspectos.
- Proporciona puntos de referencia que permiten sincronizar el vídeo con el texto, los gráficos y la animación.
- Proporciona una previsualización dinámica de las personalizaciones.
- Mantiene un tamaño de archivo SWF razonable.

El componente FLVPlayback es una combinación del área de visualización, o reproductor de vídeo, donde se visualiza el archivo FLV y los controles que permiten que funcione. Los componentes de interfaz de usuario personalizados de reproducción FLV proporcionan botones de control y mecanismos que puede utilizar para reproducir, detener, pausar o controlar de alguna otra forma el archivo FLV

De todo lo que ofrece el componente FLVPlayback, lo que más interesa es el uso de los cuepoints o puntos de referencia [V3-V6], para crear interactividad con el video

3.2.2. Cuepoints

Un punto de referencia es un punto en el que el reproductor de vídeo distribuye un evento cuePoint mientras se reproduce un archivo de vídeo. Se pueden añadir puntos de referencia a un archivo FLV cuando se desee interactuar con otro elemento de la aplicación.

Las opciones son infinitas, puede servir para mostrar texto o un gráfico, sincronizar con una animación de Flash, o pausar la reproducción del archivo FLV, buscar otro punto distinto del vídeo o cambiar a otro archivo FLV. Es decir, los puntos de referencia permiten recibir el control con el código ActionScript para sincronizar dichos puntos del archivo FLV con otras acciones de la aplicación.

Hay tres tipos de puntos de referencia: **navegación**, **eventos** y **ActionScript**. Los puntos de referencia de navegación y eventos se denominan también puntos de referencia *incorporados* porque se incorporan en el flujo de archivos FLV y en el paquete de metadatos del

archivo FLV. Se pueden incorporar desde Adobe Media Encode o tambien con Adobe Premiere Pro CS5

- **Cuepoint de navegación:** permite buscar un determinado fotograma en el archivo FLV, ya que crea un fotograma clave en el archivo FLV, lo más cerca posible al tiempo especificado. Cuando se busca un punto de referencia de navegación, el componente busca el fotograma clave e inicia el evento cuePoint.

- **Cuepont de Evento***:* permite sincronizar un instante específico del archivo FLV con un evento externo de la página Web. El evento cuePoint se produce precisamente en el instante especificado.

- **Cuepoint de ActionScript:** es un punto de referencia externo que puede añadirse a través del cuadro de diálogo Puntos de referencia de Flash Video del componente o a través del método *FLVPlayback.addASCuePoint()*. El componente almacena y rastrea los puntos de referencia de ActionScript independientemente del archivo FLV, por lo que son menos precisos que los puntos de referencia incorporados. La precisión de los puntos de referencia de ActionScript es de una décima de segundo.

Para incluir un personaje real en la aventura, se tendrá que grabar, para ello se usará que usar la técnica de Chroma Key **[C1-C6]** Esta técnica sirve para eliminar el fondo mediante software, para que quede únicamente el personaje y sustituir ese fondo por otro que en este caso se creará en Flash. Para ello se estudiará la forma de realizar un Chroma Key y el software para manipularlo. Se consideró que el mejor programa para llevar a cabo esta técnica, es el Adobe Premiere Pro CS5. Después de este proceso se deberá incluir este vídeo en el entorno de Flash.

Adobe Premiere Pro CS5

Es una herramienta de captura y edición de vídeo digital, con la que pueden conseguirse resultados profesionales.

Es capaz de capturar vídeo desde cualquier fuente: cámara digital, capturadora de televisión, memoria flash, etc y con cualquier formato, desde DVD hasta HD (Alta Definición). Edita el vídeo de forma virtual y profesional, con cientos de opciones y posibilidades.

Adobe Premiere Pro se integra totalmente con otras aplicaciones de Adobe. Esto es muy importante ya que se estará trabajando continuamente a la par, con Adobe Flash CS5 y Adobe Premiere CS5

Las características que caben destacar son:

- El manejo del chroma. Ha mejorado en Adobe Premiere CS5 gracias a la herramienta Ultra Key, capaz de detectar con precisión la clave de color incluso en situaciones de iluminación irregular.

- Permite añadir marcadores cuepoints de Flash del tipo evento o navegación a la línea de tiempo y exportar las películas directamente a formatos FLV y F4V.

- Se puede elegir entre distintos ajustes preestablecidos de Ajustes de exportación. Estos ajustes preestablecidos equilibran el tamaño de archivo frente a la calidad de audio y vídeo para obtener la velocidad de bits necesaria para cualquier

dispositivo o audiencia de destino. Si exporta la película con un canal alfa, la puede utilizar con facilidad como una capa en un proyecto.

- Puede importar el archivo FLV o F4V en Adobe Flash. Flash lee marcadores de secuencia como puntos de referencia de navegación o de eventos.

Por lo tanto, una vez analizadas las herramientas que se van a usar se puede comprobar. Es decir, después de grabar al personaje, habrá que editarlo con Adobe Premiere Pro CS5, que permite incluir los cuepoints necesarios y exportar los vídeos en formato FLV con transparencia para importarlos a Flash. Una vez estén en el entorno de Flash, se usarán las herramientas que ofrece Actionscript 3.0 para manipular los cuepoints. Para lograr todo esto, lo primero que hay que hacer es grabar los vídeos. Estas grabaciones se realizarán usando la técnica chroma key. En el capítulo siguiente se explica como realizar esta técnica

Capítulo 4: TÉCNICA CHROMA KEY

En el presente capítulo se describe la técnica del chroma key. Se definirá y se describirán los puntos principales que hay que tener en cuenta para realizarla correctamente.

La técnica del chroma key consiste en grabar a un sujeto delante de un fondo verde o azul, para, posteriormente, en la fase de edición de vídeo, eliminar ese fondo con la herramienta que el programa de edición que se use lo permita, en este caso, se utilizará la herramienta *Ultra key* de Adobe Premiere Pro CS5. De esta forma, quedará únicamente el personaje y se podrá colocar el fondo que se necesite

Para realizar un buen chroma key, se tiene que tener en cuenta tres aspectos fundamentales: el fondo, la iluminación y la cámara. A continuación, se explicará cuales son los puntos más importantes a tener en cuenta de estos tres aspectos.

4.1 EL FONDO.

Suele ser azul o verde, debido a que son los colores que están menos presentes en la piel de las personas. Es recomendable que los colores sean puros, es decir, que no contengan nada de sus otros dos componentes de los colores primarios.

La tela del fondo no debe tener arrugas para evitar variaciones de luz y la existencia de sombras. Lo más importante es que el fondo este lo más uniformemente iluminado.

Si el fondo es demasiado brillante, se tiene que tener cuidado, a la hora de iluminarlo ya que se pueden "ensuciar" los bordes del sujeto con el color del fondo.

4.1.1. Iluminación:

Debe haber una diferencia significativa entre la luminancia del sujeto y la del fondo **[C4].** El fondo se debe iluminar con un nivel más bajo que el primer plano, con esto se evitará que el color del fondo se refleje en el sujeto.

Se puede iluminar el fondo de diversas formas y con diferentes tipos de focos. A continuación, se explicarán dos posibles formas de hacerlo:

- Una de ellas consiste en usar dos focos y papel difusor que se coloca delante de éstos. Los focos se sitúan a cada lado del fondo a unos 15º del mismo. El foco de la izquierda apuntará a la parte derecha y viceversa [01].

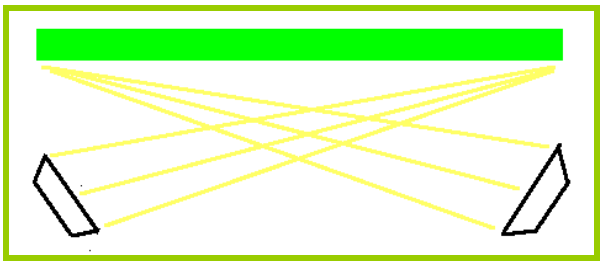


Figura 4.1 Iluminación cruzada

- Otra opción es colocar paneles difusores a cada lado del fondo inclinados de tal forma que al hacer rebotar la luz, esta ilumine el lado contrario donde están colocados.

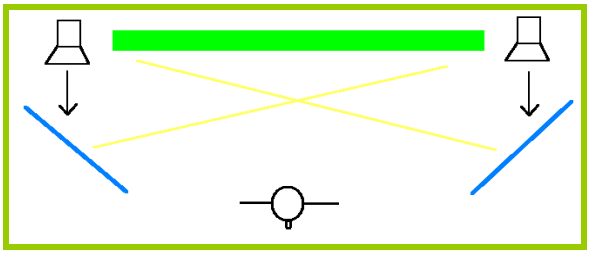


Figura 4.2 *Iluminación con paneles difusores*

Como se ha dicho existen muchas formas de iluminar un fondo y los ejemplos anteriores son solo dos de todas ellas.

El plató de televisión de la universidad, donde se realizarán las grabaciones para este proyecto, dispone de 4 fluorescentes calibrados a 3200K, que cubren el fondo por completo. Los fluorescentes dan una luz difusa, por lo que no hace falta el uso de paneles difusores.

4.2 EL ACTOR.

El aspecto más importante es que deberá llevar ropa de distinto color al fondo. Lo más aconsejable es que la ropa sea de un color complementario al del fondo.

Debe situarse lo más lejos posible del fondo para evitar que se creen sombras indeseadas en éste. Se considera que un metro es una buena distancia para evitar este tipo de problemas [C3].

4.2.1. Iluminación:

Para iluminar al actor se puede utilizar la técnica de iluminación a tres puntos **[02].** En este tipo de iluminación, se utilizan dos fluorescentes, para la luz principal y la luz de relleno y un fresnel con un filtro para el contraluz.

La **luz principal** se ubica 45 º a la derecha o izquierda del sujeto con cerca de 45º de inclinación hacia abajo, siempre dirigiéndola hacia la cara.

La **luz de relleno** va situada al lado contrario de la principal y a una altura similar a la de la cámara, con una intensidad menor a la de la principal ya que su objetivo es modelar el rostro mediante la creación de sombras y contornos y así evitar una apariencia plana. Se recomienda que su posición sea unos 15º a 25º del sujeto.

Por último está la luz de atrás y por encima del sujeto, el **contraluz**, que se usa para separar el sujeto del fondo. Para los contraluces se pueden usar filtros de colores complementarios al fondo, en este caso, como el fondo es verde, se usarán filtros de color magenta. Así se pueden neutralizar los reflejos verdes que pueden aparecer en el contorno del sujeto, pero se debe tener cuidado con esta técnica ya que si no se realiza correctamente se corre el riesgo de contaminar al actor con el color del filtro.

Para finalizar, es importante destacar que la iluminación del fondo y la del sujeto debe tener la misma temperatura de color.

4.3. LA CÁMARA.

A la hora de grabar también se tiene que tener en cuenta ciertos aspectos, no solo basta con una buena iluminación del fondo y de la persona.

Cuando se vaya a hacer el balance de blancos, se debe hacer en un plano cerrado ya que, de lo contrario, pueden colarse reflejos de luz de otros colores y así obtener un valor incorrecto de la temperatura de color.

Se deberá tener perfectamente enfocado al sujeto y si es posible, desenfocar el fondo. Además, se deberá saber como controlar la profundidad de campo en el caso de que el personaje cambie su posición y pueda salirse de la zona enfocada. Hay tres factores que influyen en la profundidad de campo **[C6]**:

- **Distancia de enfoque**: cuanto más lejos se enfoque, mayor será la profundidad de campo.

- **Distancia focal o zoom**: a mayor distancia focal (aumentar el zoom) menor será la profundidad de campo.

- **Apertura del diafragma**: Cuando menor sea la apertura del diafragma (mayor f), mayor será la profundidad de campo.

Sabiendo esto, se tendrá que abrir el diafragma al máximo (número f más pequeño) enfocar a nuestro primer plano y tener la distancia focal más grande posible.

Otro aspecto importante es no usar las ganancias positivas de la cámara ya que aumenta el ruido en la imagen. En el caso de que la cámara disponga de ganancia negativa, es muy recomendable usarla.

Por último, se tendrá que elegir el formato de grabación que más convenga, esto lógicamente depende de la cámara que se vaya a utilizar, pero a rasgos generales, se debe saber que existen varias opciones:

Puede hacerse en DV que es, resumidamente, un sistema de vídeo digital que utiliza un factor de submuestreo de croma de 4:2:0 en PAL y una resolución de 720x576. También se tiene la opción de grabar en HDV que es un formato de vídeo que usa las propias cintas DV, pero consiguiendo una mayor calidad, ya que graba a 1440x1080 con una mayor compresión, esto es debido a que usa un estándar de compresión MPEG-2. Es lógico que cuanta mayor calidad en la imagen, mejores resultados se obtendrán en la edición del chroma, pero en ocasiones no es conveniente utilizar la máxima calidad de vídeo, si eso implica, ralentizar el proceso de edición o incluso que el propio software no pueda trabajar con esos formatos. En todo caso, se debe saber con anterioridad, que es lo que se necesita para optimizar al máximo el trabajo.

La cámara que se va a usar para las grabaciones es la Sony PMW EX3 que ofrece diversos formatos. Para poder determinar qué se necesita y con qué se puede trabajar, se realizarán una serie de grabaciones que se detallarán más adelante y que determinarán la elección del formato de vídeo que va a usarse finalmente.

Capítulo 5: ENSAYOS INICIALES

La fase más larga y costosa del proyecto fue en la que se realizaron una serie de pruebas y ejercicios, que abarcaban todos los aspectos necesarios para crear la aventura final. También fue la parte más importante, ya que junto con el estudio previo que se realizó antes de comenzar el proyecto, supuso todo el aprendizaje necesario para ir avanzando en su desarrollo.

Estos ensayos pretenden profundizar en todo lo relacionado con: la programación en ActionScrip 3.0, la manipulación de vídeo en Flash, el manejo de una cámara de vídeo, la grabación de un personaje mediante la técnica Chroma key y la edición y exportación de los vídeos.

Las ensayos se presentan en el orden en que fueron realizadas. En total se hicieron cinco tipos, que se explicarán uno por uno más adelante, estas son:

-Integración de vídeo en Flash y uso de cuepoints.
-Creación de pequeñas aplicaciones en Flash.
-Manejo de la cámara.
-Manejo de la iluminación y edición de los vídeos.
-Grabación de diferentes encuadres del personaje.

Para todos ellos, se siguió el mismo proceso. Primero, se describió lo que se quería conseguir con cada uno y las partes y ejercicios de los que constara. A continuación, se procedió a la realización de la prueba en cuestión y por último se analizaron los resultados obtenidos y se sacaron las conclusiones oportunas para así poder tomar las decisiones finales para crear la aventura gráfica.

La primera prueba que se realizó se explica a continuación.

5.1. INTEGRACIÓN DE VIDEO Y USO DE CUEPOINTS.

Para comprobar que se podía integrar vídeo real con transparencia en flash e interactuar con el a través de unos botones programados en ActionScript 3.0, se realizó esta sencilla prueba. Se hizo sin profundizar en los parámetros de edición de chroma y de exportación de vídeo, ya que más adelante se hablará de cada uno de ellos, de forma detallada.

El vídeo se grabó con una cámara de fotos compacta y el fondo verde, se conseguirá con una manta de ese color. El objetivo no es obtener buena calidad en el vídeo sino, determinar los pasos a seguir para las demás pruebas y grabaciones que se realizaran más adelante y decidir la forma en que se va a trabajar.

Los pasos que se siguieron para realizar esta prueba fueron: primero grabar a la persona sobre un fondo verde, después eliminar ese fondo verde mediante la herramienta Ultra key de Adobe Premiere Pro CS5, a continuación exportar el vídeo con canal alpha [V7] para poder ser visualizado en Flash y por último añadir el código necesario para la manipulación del vídeo a través de los cuepoints. A continuación, se explicará cada paso por separado.

5.1.1. Grabación de vídeo con fondo verde:

El primer paso es grabar a una persona con un fondo verde detrás. La persona realizará tres tipos de movimientos, cada movimiento lo hará durante unos diez segundos. Los cuepoint que se añadirán más adelante, estarán colocados en el inicio de cada movimiento, para que a la hora de interactuar con el vídeo, se puedan apreciar más fácilmente los cambios.

El vídeo está grabado con una cámara de fotos Nikon COOLPIX L16.

Para el fondo verde se usó una manta verde.

Figura 5.1. Vídeo grabado con cámara compacta con fondo verde

5.1.2. Eliminar fondo verde en Premiere

Para eliminar el fondo se usará el programa Adobe Premiere Pro CS5. El chroma se eliminará utilizando el efecto Ultra key, situado, en efectos de vídeo, en la carpeta Keying.

Como ya se ha mencionado antes, en este capítulo no se profundizará en la forma de eliminar el chroma, ni en los parámetros que lo controlan ya que no es el objetivo de esta prueba, solo con conseguir un resultado aceptable, para comprobar que se puede exportar con el fondo transparente para visualizarlo en flash, es suficiente. Más adelante se dedicarán unas pruebas exclusivamente a este tema, donde se detallará todo este proceso y se explicará la función de cada uno de los parámetros.

El proceso a seguir es el siguiente:

Con la herramienta cuentagotas se elige el color del fondo para que se elimine. A continuación, se ajustan los niveles de los diferentes parámetros, hasta tener el fondo verde eliminado por completo.

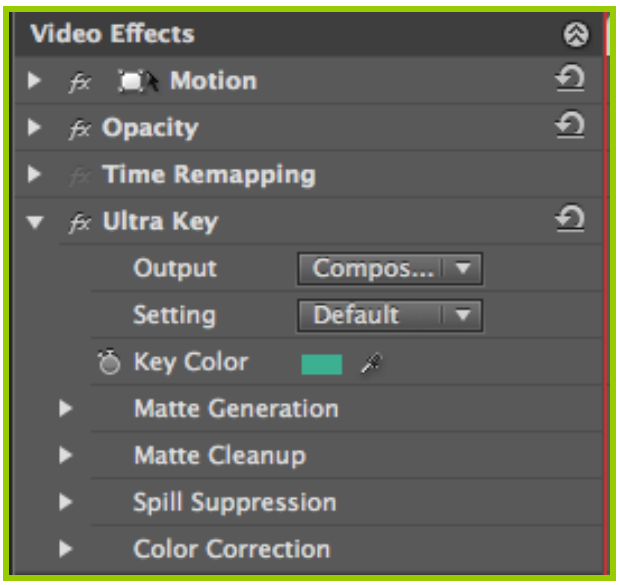


Figura 5.2. Parámetros del efecto Ultra key

5.1.3. Exportación de vídeo e integración de cuepoints:

Una vez se tenga el vídeo con el fondo eliminado, se deberá exportar para que pueda ser visualizado en flash.

Para ello se elige **File** > **Export** > **Media**

En el menú de exportación, *figura 5.3.*, se puede ver que está dividido en dos partes, a la izquierda, aparece el vídeo y debajo de él, las opciones de agregar o eliminar puntos de referencia, del tipo navegación o de evento y en la parte derecha aparecen las opciones de exportación del vídeo.

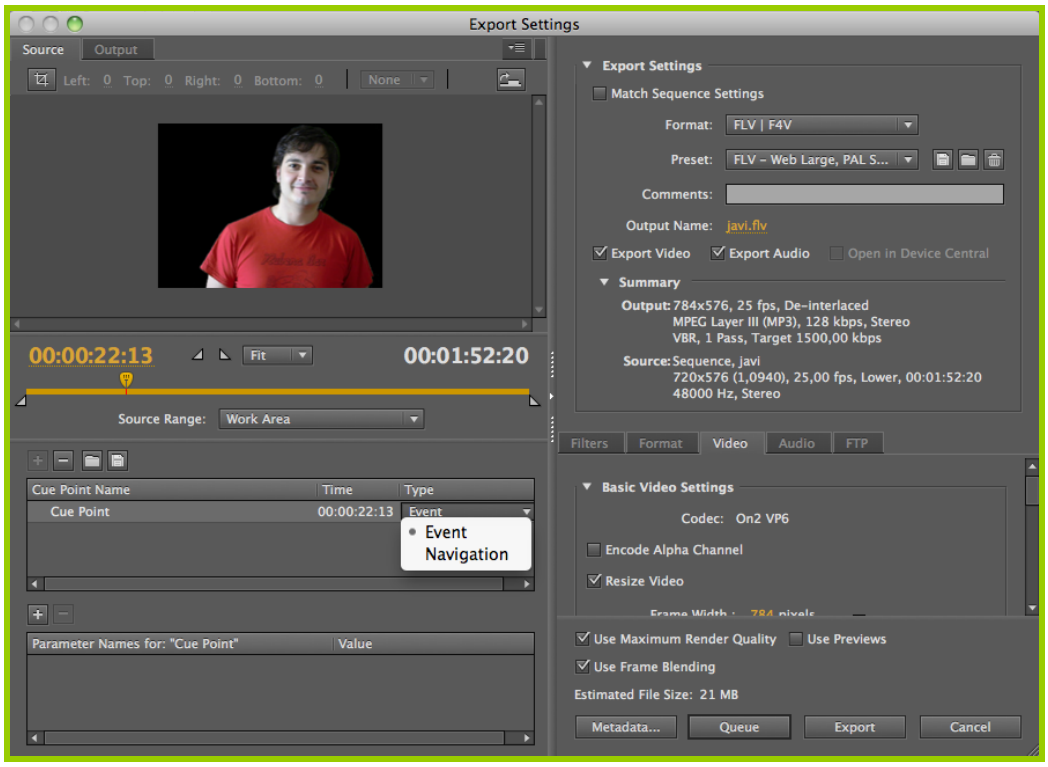


Figura 5.3. Ventana de los ajustes de exportación

Lo primero que se hará es añadir los puntos de referencia, para ello, basta con colocarse en el punto de la línea de tiempos que se desea y pulsar el botón , se añadirá un cuepoint. Se podrá editar el nombre (para poder llamarlo desde ActionScript 3.0) y de que tipo es, si de navegación o evento. En este caso, interesa que sea de navegación, ya que el objetivo es que salte de un punto a otro del vídeo, en función del botón que se pulse.

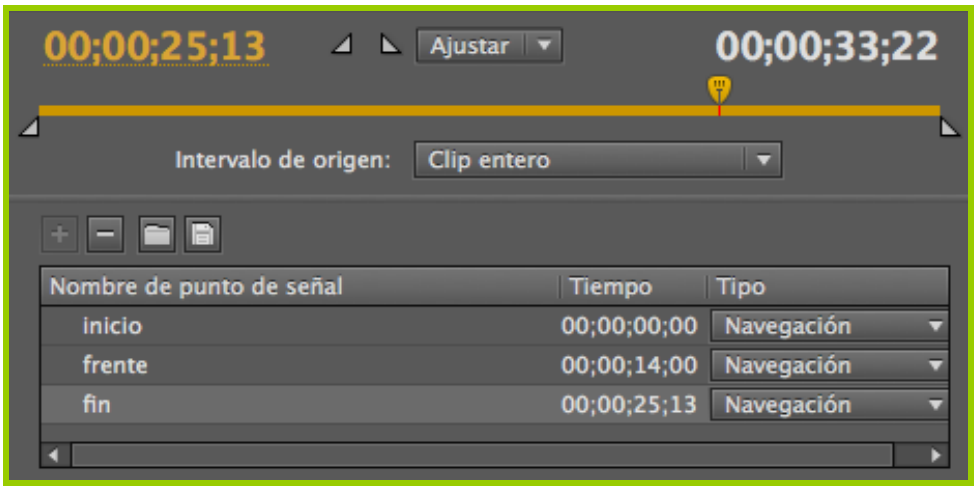


Figura 5.4.Detalle de la ventana de incrustación de los cuepoints

Una vez añadidos los cuepoint, ya se pueden elegir los parámetros de exportación de vídeo. Lo básico para que se pueda exportar el fondo transparente y visualizarlo en flash es:

- Elegir el formato **FLV**, ya que permite la codificación del canal alfa.

- En ajustes básicos de vídeo, se elige **Codificar el canal alfa**. Esto permite codificar el fondo que se ha eliminado, que en esencia, es transparente.

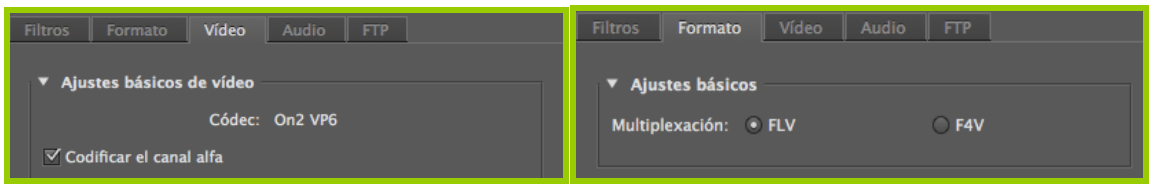


Figura 5.5. Ventanas de formato y códec de vídeo

5.1.4. Código para el control de la película:

Una vez exportado el vídeo, ya puede ser importado para visualizarse en flash. Para ello, se abre Flash y en **File > Import > Import to Stage** se elige el vídeo que se acaba de editar.

Antes de aparecer en el escenario aparecerán varias opciones de importación y visualización del vídeo, como el que se ve en la *figura 5.6.*:

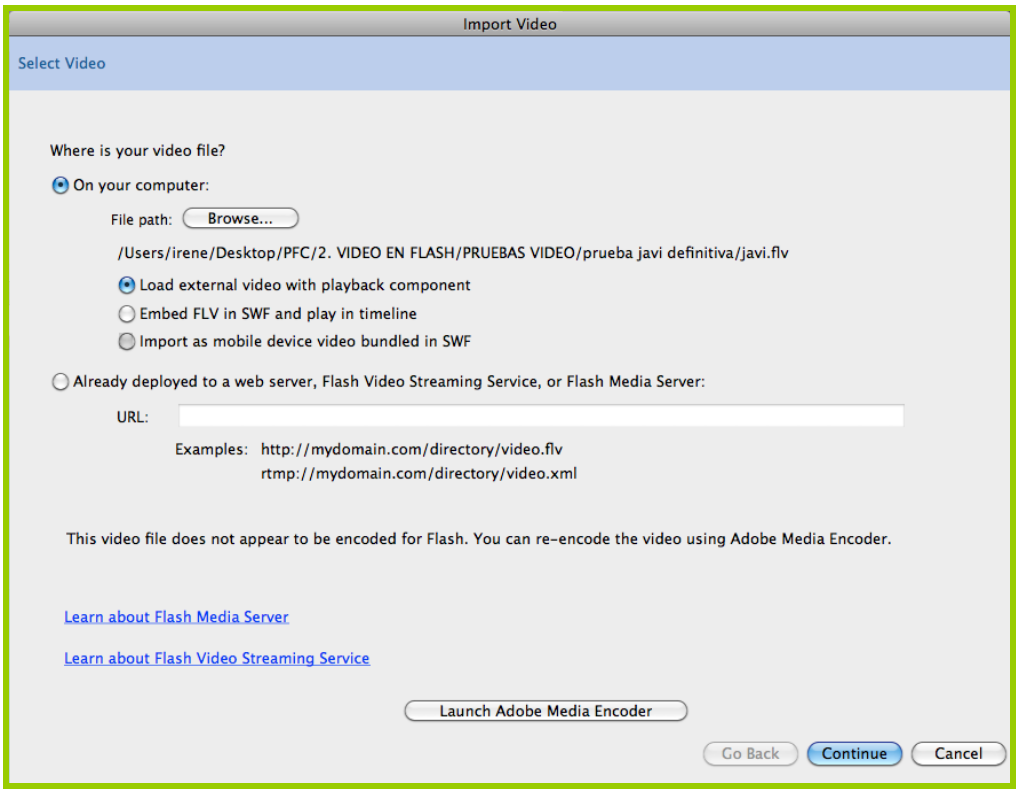


Figura 5.6. Opciones importación del vídeo en Flash

Se elegirá la primera, **Load external video with playback component**, ya que permitirá programar el vídeo mediante ActionScript 3.0 y aprovechar las propiedades que ofrece el componente FLVplayback, como la de integrar y programar puntos de referencia.

A continuación, hay que elegir el tipo de visualización que se quiere para el vídeo, (el formato de los botones, el volumen, etc.) se elegirá **None,** como se ve en la *figura 5.7.*, ya que no interesa que aparezca ningún menú

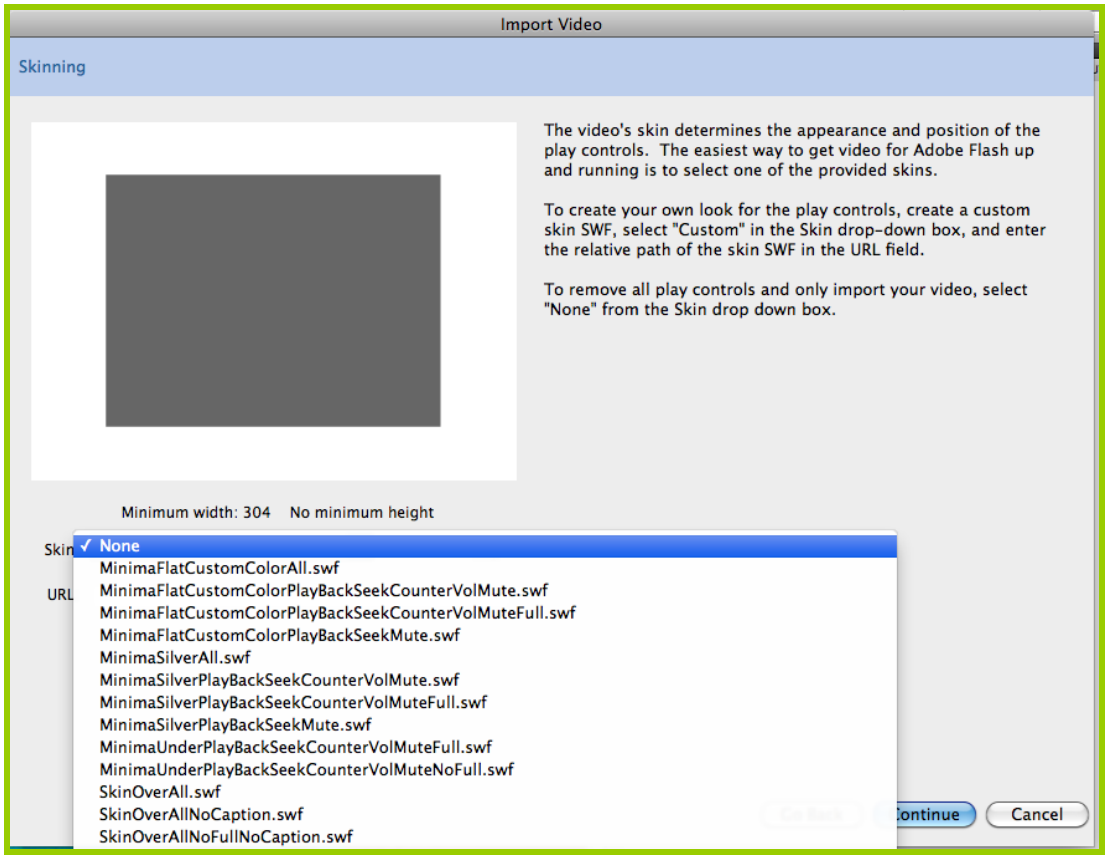


Figura 5.7. Opciones de visualización del archivo de vídeo

Una vez se tiene el vídeo en el escenario, lo primero que se tendrá que hacer es darle un nombre, para ello hay que acceder al panel propiedades de flash, en este caso se ha llamado **mivideo.**

A continuación habrá que añadir la programación, lo que se quiere conseguir es:

- **Programar** una serie de **botones**:
    - Avance (pasar al siguiente cuepoint de navegación)
    - Retroceso (ir al anterior cuepoint de navegación)
    - Pausa (pausar el video)
    - Stop (parar el video y volver al principio)
    - Play (reanudar el video)

- -Hacer **bucle en el primer movimiento** si no se toca ningún botón.

- -Cada vez que se cambie de movimiento, **cambiar el fondo de la imagen.**

El código con los comentarios necesarios de nuestra película es el siguiente. En la carpeta del DVD **Pruebas > 5.1 IINTEGRACION DE VIDEO Y USO DE CUEPOINTS**, se encuentran los ficheros de flash: **javi.fla**, el vídeo con el fondo eliminado: **javi.flv** y el proyecto de Adobe Premiere Pro CS5: **javi.prproj.**

Lo primero que hay que hacer es importar las clases que se necesitarán, para visualizar el vídeo y controlarlo a través de los cuepoint.

También se añadirá un cuepoint, para detectar el final del primer movimiento. La última parte hará que se muestre en la salida, información sobre el cuepoint que se detecte.

```
//Importo las clases necesarias.

import fl.video.*;
import fl.video.MetadataEvent;
import flash.sampler.pauseSampling;

//Se añade este punto de referencia tipo actionscript
//para poder realizar el bucle.
mivideo.addASCuePoint(13.9, "bucle");


//Esto da la información de los cuepoints que hay, cada vez
//que se pasa por ellos.
mivideo.addEventListener(MetadataEvent.CUE_POINT, informacion);
function informacion(eventObject:MetadataEvent):void {
        trace(mivideo.playheadTime + "segundos");
        trace("Nombre del evento: " + eventObject.info.name);
        trace("Tipo de evento:" + eventObject.info.type);
}
```

Esta es la parte del código donde se programan los botones. Previamente, se han tenido que dibujar los botones, convertirlos a movieclip y añadirles un nombre de instancia en el panel propiedades para poder llamarlos desde ActonScript 3.0.

```
//Se programan los botones retroceder, avanzar, play, stop y //pause.
btnretroceder.addEventListener(MouseEvent.CLICK,retroceder);

function retroceder(event:MouseEvent){
      mivideo.seekToPrevNavCuePoint();
}

btnavanzar.addEventListener(MouseEvent.CLICK,avanzar);

function avanzar(event:MouseEvent){
      mivideo.seekToNextNavCuePoint();
}

btnplay.addEventListener(MouseEvent.CLICK, empezar);

function empezar(event:MouseEvent){
      mivideo.play();
}

//Al hacer stop, vuelva al inicio
mivideo.autoRewind=true;

btnstop.addEventListener(MouseEvent.CLICK,parar);

function parar(event:MouseEvent){
      mivideo.stop();
}

btnpause.addEventListener(MouseEvent.CLICK, pausar);

function pausar(event:MouseEvent){
      mivideo.pause();
}
```

En esta parte, el código hace el bucle en el primer movimiento.

```
//Esta es la parte del código que hace el bucle.

/*Lo primero que hace es detectar el punto de referencia que indica
el final del primer movimiento, en este caso se llama "bucle".*/
/*Cuando lo detecta salta al cuepoint "inicio" colocado al
//principio del vídeo.*/

mivideo.addEventListener(MetadataEvent.CUE_POINT, bucle);

function bucle(eventObject:MetadataEvent):void {
    if(eventObject.info.name == "bucle")
        mivideo.seekToNavCuePoint("inicio");
      }
}
```

La última parte del código controla los fondos, cada vez que se cambia de movimiento, cambia el fondo. Cada fondo está asociado a un cuepoint y cada vez que se detecta el cuepoint se visualiza a el fondo asociado. Cada fondo está colocado en un fotograma diferente.

```
//Esta parte del código es la que controla los fondos.

//Al detectarse el cuepoint, se ordena ir al fotograma donde se
//encuentra el fondo que se quiere visualizar.

mivideo.addEventListener(MetadataEvent.CUE_POINT, agua);
            function agua(eventObject:MetadataEvent):void {
      if(eventObject.info.name == "inicio")
        gotoAndStop(1);
      }
}

mivideo.addEventListener(MetadataEvent.CUE_POINT, palmeras);
function palmeras(eventObject:MetadataEvent):void {
      if(eventObject.info.name == "frente")
        gotoAndStop(2);
      }
}

mivideo.addEventListener(MetadataEvent.CUE_POINT, nieve);

function nieve(eventObject:MetadataEvent):void {
      if(eventObject.info.name == "fin")
        gotoAndStop(3);
      }
}
```

5.2. CREACIÓN DE APLICACIONES EN FLASH.

Una vez comprobado que puede incluirse vídeo en Flash con transparencia y que puede manipularse de una manera muy sencilla gracias a los cuepoint, se procederá con las aplicaciones realizadas en Flash. Se elaboró una lista con todas las acciones que se creyeron necesarias para crear la aventura final. Así en el momento en el que se fuera a programar no resultara tan costoso. Además sirvió para empezar a familiarizarse con el lenguaje ActionScript 3.0. Para ello se consultaron manuales y tutoriales [F1-F43] relacionados con la programación con ActionScript 3.0 y con las pruebas en concreto. La lista de las aplicaciones que se elaboró fue en función de lo que ya se conocía sobre las aventuras gráficas: los tipos que existen, la interfaz que usan, las acciones que se pueden realizar y las formas de interactuar con los objetos. Los archivos de cada aplicación se encuentran en el DVD, en la carpeta **Pruebas > 5.2 CREACION DE APLICACIONES EN FLASH.** El nombre de cada archivo es el número que le corresponde a cada aplicación en esta memoria.

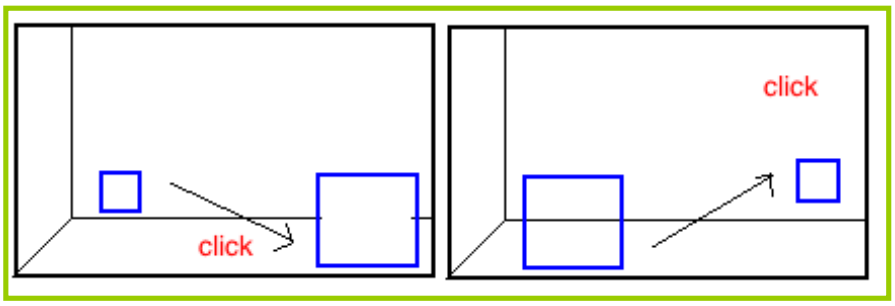
Cabe decir, que todas las pruebas no se hicieron al mismo tiempo. Las que implican únicamente objetos creados en Flash se hicieron las primeras, pero las que contienen vídeo se realizaron más tarde debido a que se tenían que realizar las pruebas de vídeo previas. Además una vez realizadas las pruebas de vídeo de los gestos de personaje, se vio que en ocasiones, no se accedía al cuepoint que se le indicaba y realizaba otro gesto. Para solucionar esto se describió otra prueba, para comprender mejor el funcionamiento de los cuepoints. Gracias a esta prueba pudo solucionarse el problema y por último se incluye la aplicación definitiva con el correcto funcionamiento de los cuepoints.

A continuación, se describe la lista que se elaboró, con todas las aplicaciones. Incluye un pequeño resumen de lo que hará y vendrá acompañado con dibujos para hacerlo más visual. Después se procederá a explicar detalladamente una por una cada aplicación.

Crear perspectiva real

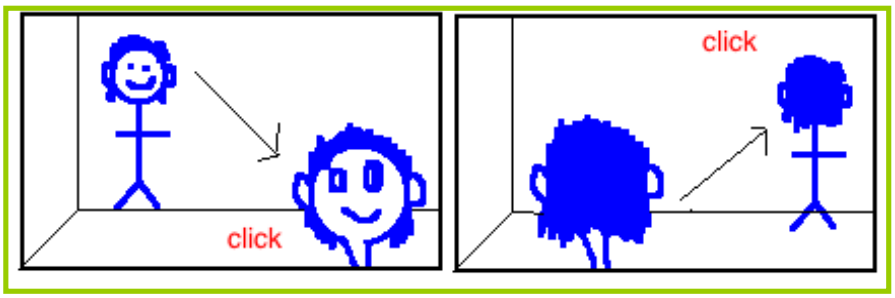
➢ PERSPECTIVA CON MOVIECLIP

Al hacer click en la parte superior del escenario el objeto avanzará hasta esa posición disminuyendo a la vez su tamaño y al pinchar en la parte inferior del escenario, avanzará hasta esa posición aumentando su tamaño.



➢ PERSPECTIVA CON VÍDEO

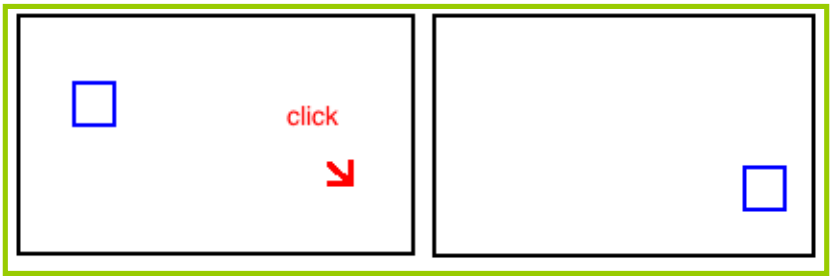
Una vez se tenga el vídeo grabado, se hará lo mismo que en el punto anterior.



Movimiento de objetos

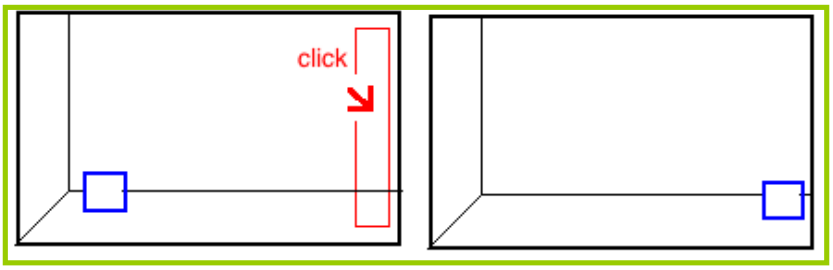
➢ TRASLADAR A UN PUNTO

Al hacer click con el ratón en un punto del escenario, el objeto se mueva hasta ese punto.



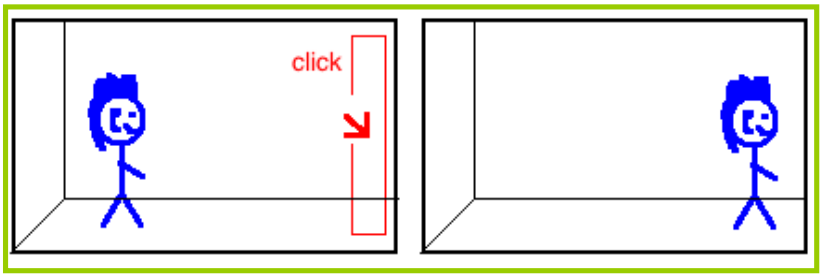
➢ MOVER MOVIECLIP

Al hacer click en la parte izquierda o derecha del escenario el objeto se moverá hasta la parte izquierda o derecha, pero solo cambiará la posición en x, no en y.



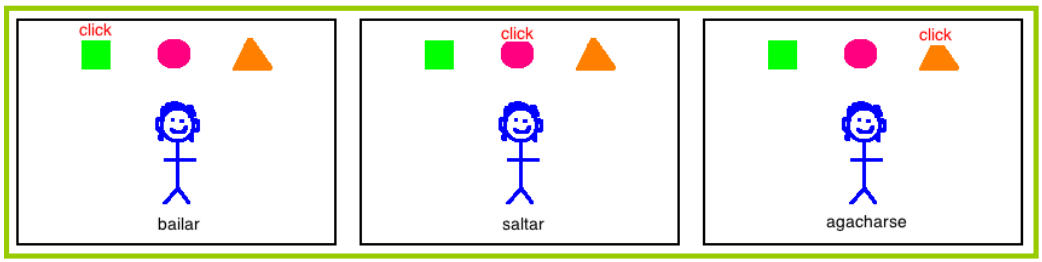
➢ MOVER VÍDEO

Se hará lo mismo que en el punto anterior pero con el vídeo grabado



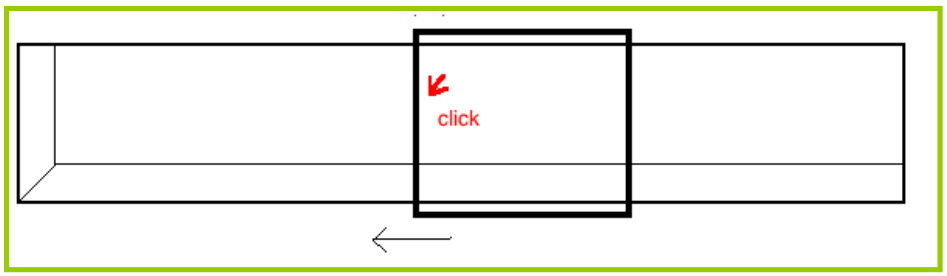
Cambiar gestos de los vídeos

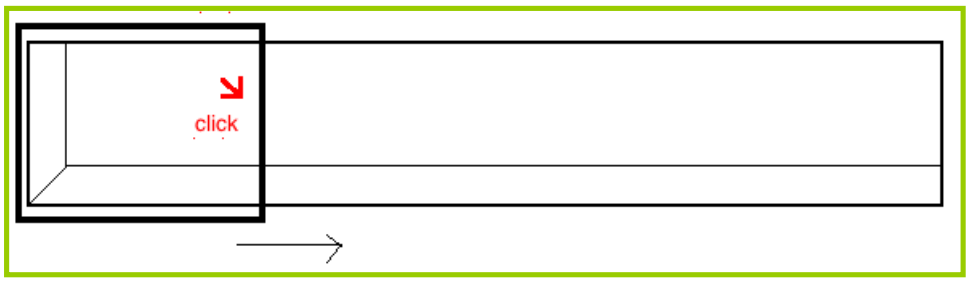
Después de grabar el vídeo de la persona haciendo diferentes gestos, dependiendo de que objeto vaya a coger hará un gesto u otro.



Mover el escenario

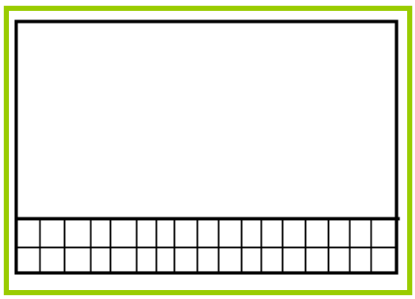
Teniendo un escenario más grande que la ventana de flash, al hacer click en la parte izquierda o derecha del escenario este avanzará un poco, dejando ver más parte del escenario.





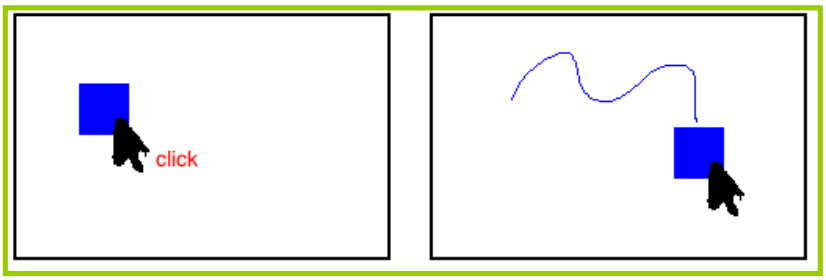
Crear lugar para almacenar los objetos

Se creará un lugar en la parte inferior o superior del escenario donde puedan almacenarse los objetos, un inventario.



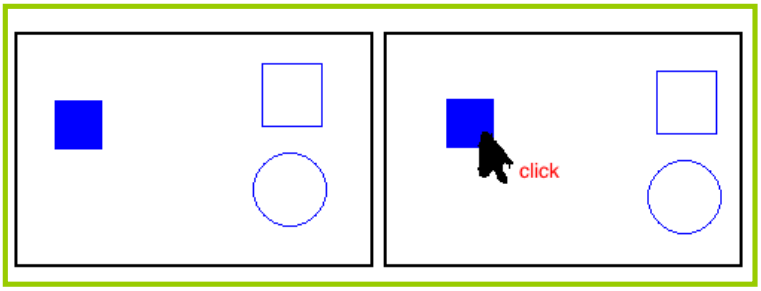
Coger objetos y arrastrar

Se colocará un objeto y al hacer click con el ratón se podrá arrastrar.

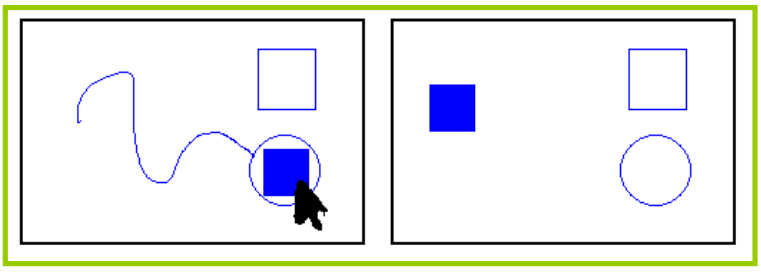


Coger objetos, arrastrar y dejar en lugar determinado

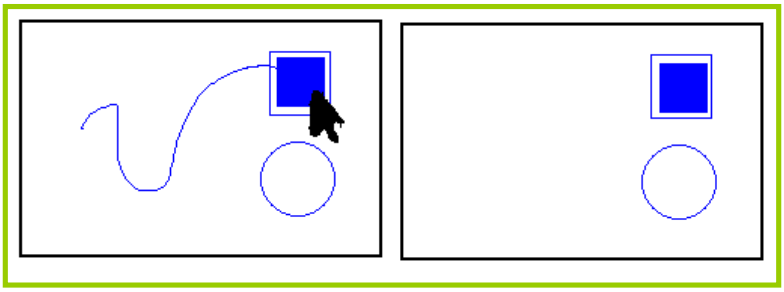
Después de lograr arrastrar un objeto, se tendrá que poder dejarlo en un lugar determinado, no en cualquier sitio. Si se coloca en el lugar equivocado volverá a su posición inicial.



Si no acierta vuelve a su posición:

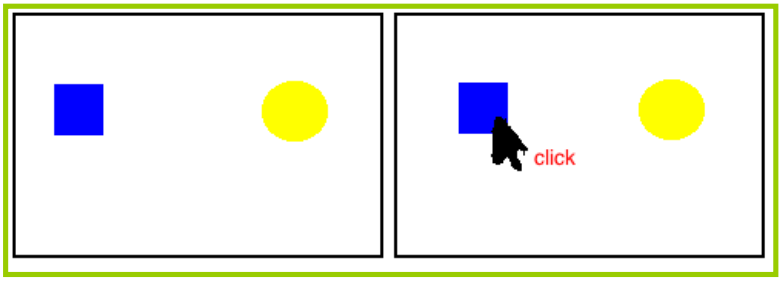


Si acierta se queda en el sitio:

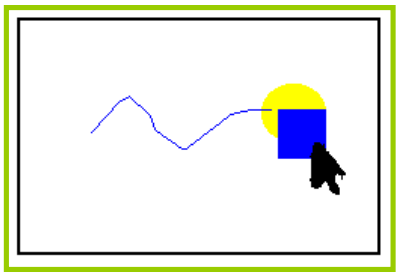


Combinar objetos

Al arrastrar un objeto hasta otro, estos se combinan, formando un objeto nuevo.



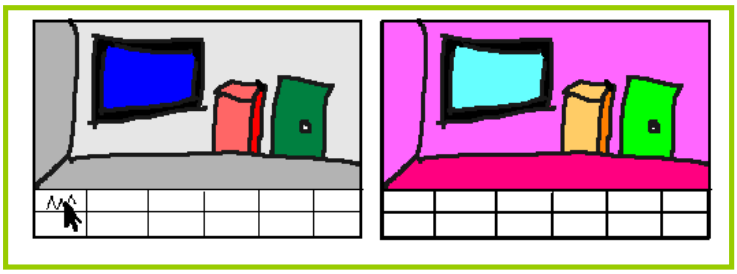
Se arrastra el cuadrado hasta el círculo:



Cambia a otra forma verde:

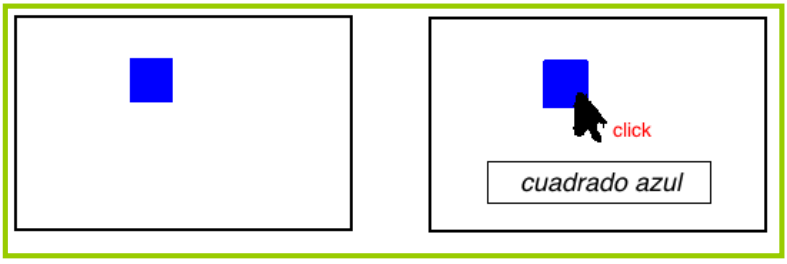
Cambiar el escenario

Al coger un objeto determinado del inventario, el escenario cambia.



Aparecer texto

Al hacer click sobre algo aparece texto.

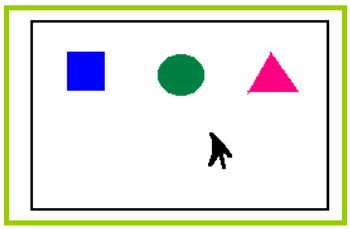


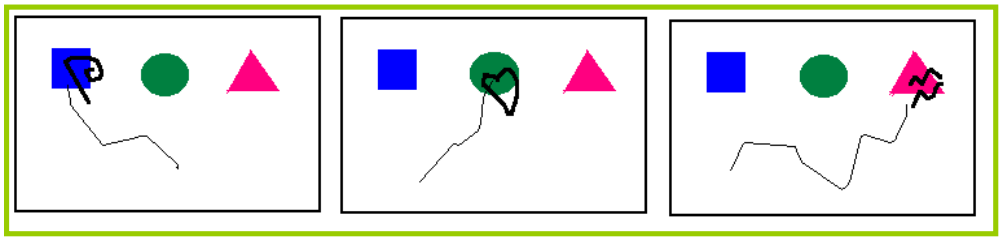
Condiciones para que ocurran cosas

Al hacer click sobre algo, si se tiene un objeto aparecerá un texto y si no se tiene, aparecerá un texto diferente.

Cambiar el cursor

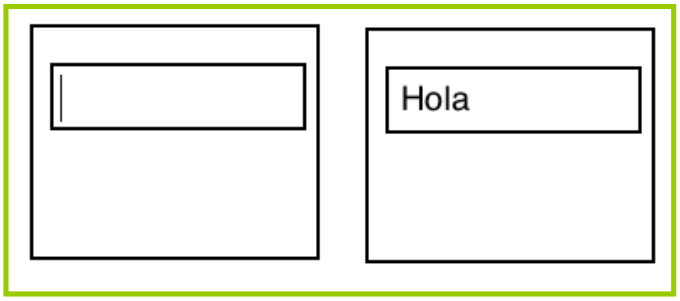
En el mismo escenario, dependiendo por donde se pase el cursor este cambiará de aspecto.





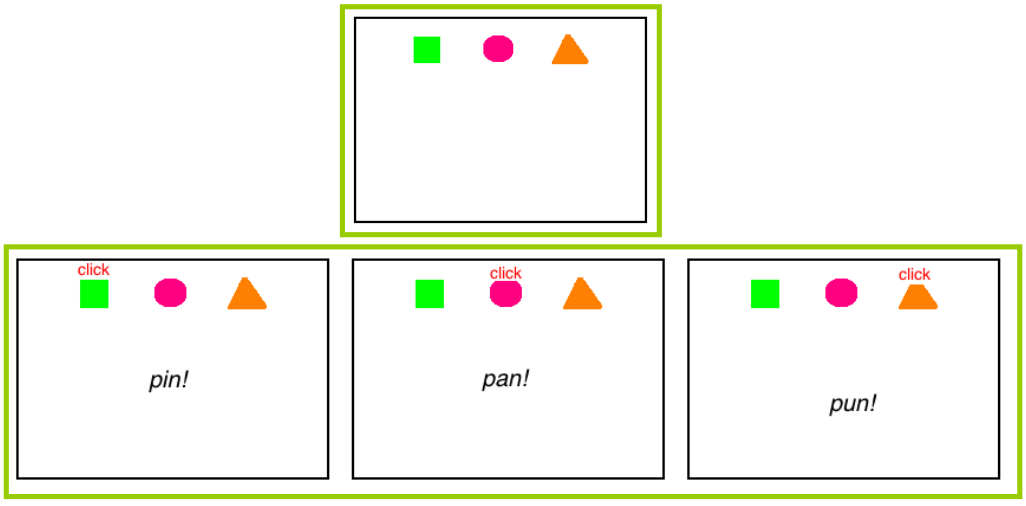
Introducir el texto

Poder meter texto en un lugar determinado.



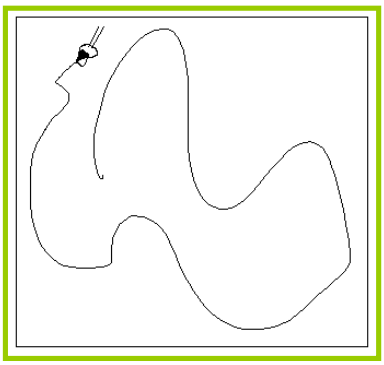
Reproducir sonidos

Depende del objeto que se pulse, se oirá un sonido u otro.



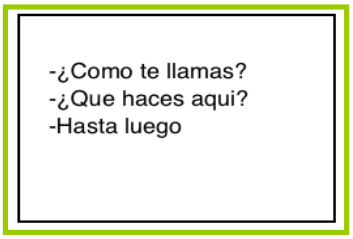
Dibujar

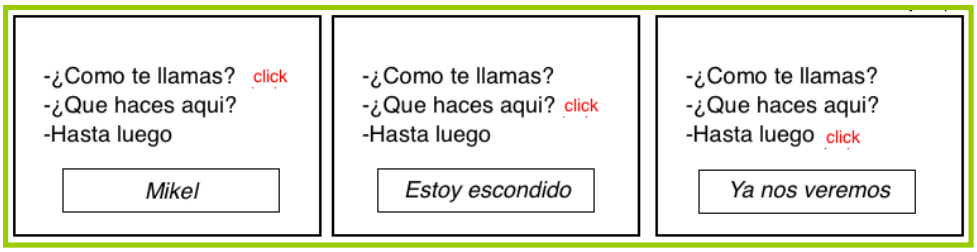
Aplicación con la que se puede dibujar como si fuera el Paint.



Diálogos

Crear una serie de preguntas y dependiendo cual se elija, se obtendrá una respuesta u otra.





Una vez elaborada y descrita la lista se prosiguió a la realización de cada aplicación. Antes se estudiaron los aspectos y términos básicos de ActionScript 3.0.

Para desarrollar estas aplicaciones se utilizaron una serie de eventos y métodos que son comunes a muchas de ellas [F1]. A continuación, se explicarán con más detalle los más importantes.

➢ Eventos:

Los eventos son algo que le sucede a un objeto. Cada evento se representa mediante un objeto de evento, que es una instancia de la clase Event o de alguna de sus subclases. Los objetos de evento almacenan información sobre un evento específico, y también contienen métodos para manipular los objetos de evento.

ENTERFRAME: este evento se asocia con la ejecución de la película flash, ya que la función llamada con este evento se ejecuta constantemente, es decir está relacionado con la velocidad de la película, los fps.

Eventos de ratón, estos permiten interacción con el usuario al hacer una acción con el ratón. Los que se han usado para las pruebas son:

MOUSE_DOWN: se produce al pulsar el ratón, antes de soltarlo.
MOUSE_UP: se produce cuando se suelta el botón del ratón.
MOUSE_OVER: se produce cuando se coloca el ratón encima del objeto.
MOUSE_OUT: se produce cuando se saca el ratón de encima del objeto.
CLICK: se produce al hacer click.

➢ Métodos:

Los métodos son acciones que pueden llevar a cabo los objetos. Algunos de los métodos comunes que se han utilizado para las pruebas son:

Los **detectores de eventos** o listeners: son funciones o métodos que se escriben para responder a los distintos eventos.

addEventListener(*"nombreEvento", nombreFuncion*):

Se usa para detectar un evento. En *"nombreEvento"* se escribe el evento que se quiere detectar y *nombreFuncion*, es la función que se ejecutará cuando se detecte el evento.

removeEventListener(*"nombreEvento",nombreFuncion*)

Se usa para eliminar un detector de evento. En *"nombreEvento"* es el evento que se desea eliminar, *nombreFuncion*, es la función asociada al evento.

Otros métodos usados:

startDrag(); Se usa para arrastrar un objeto con el ratón.

Parámetros de entrada→ Ninguno.
Salida→ void.

stopDrag(); Se usa para parar de arrastrar.

Parámetros de entrada→ Ninguno.
Salida→ void.

addChild(child:DisplayObject**);** Añade un objeto al escenario o a un contenedor para que pueda ser visualizado.

Parámetro de entrada→ DisplayObject: objeto que deseamos añadir a la lista de visualización.
Salida→ DisplayObject: instancia del objeto.

getChildByName(name:String**);** Devuelve el objeto que viene con la propiedad *name* especificada.

Parámetro de entrada→ String: objeto con la propiedad name.
Salida→ DisplayObject: objeto de visualización.

hitTestObject(obj:DisplayObject**);** evalúa a un objeto para ver si a colisionado con otro.

Parámetro de entrada→ DisplayObject: objeto que deseamos evaluar.
Salida→ void.

Con estas pruebas se pretende tener una idea de lo que se puede hacer o no con Flash y a la hora de implementarlo en la aventura gráfica, sea menos costoso. A continuación, se describe con detalle cada aplicación.

5.2.1. Crear perspectiva real

- CON UN MOVIECLIP

Objetivo:

El cuadrado azul situado en el escenario comenzará a acercarse y a aumentar de tamaño cuando se presione el botón verde.

Para esta prueba se necesitan dos movieClips, uno que hará la función de botón que se llamará: *pulsar* y el cuadrado azul: *mccua*. Al hacer click en *pulsar*, *mccua* irá desplazándose en el eje *y* (acercándose) y aumentando su escala de forma proporcional, para poder crear el efecto de que se acerca de forma realista.

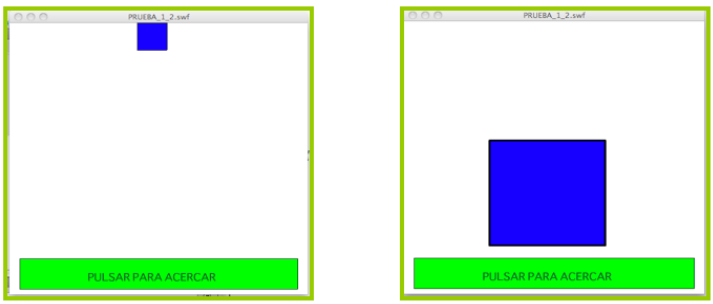


Figura 5.2.1 Estado inicial y final. Crear perspectiva con movieclip.

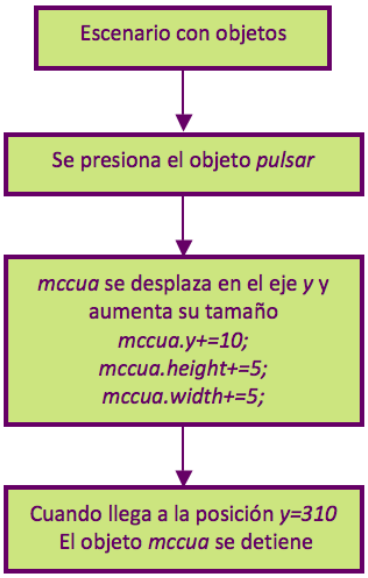


Diagrama 5.1. Prueba crear perspectiva

Código:

Inicialmente se tiene el escenario con los dos objetos, *pulsar* y *mccua.*

Se crea un evento de ratón del tipo **MouseEvent.CLICK** en el objeto *pulsar*, que se llama con el detector de eventos ya explicado **addEventListener()**, este listener tiene asociado la función **acercar().**

```
pulsar.addEventListener(MouseEvent.CLICK,acercar);
```

La función **acercar()** tiene asociado el evento **ENTERFRAME** que activa la función **ciclo()**, que será responsable del movimiento y aumento de tamaño del objeto *mccua*, irá sumando su posición vertical en 10 y su escala *x* e *y* en 5 píxeles. Al llegar a la posición *y=310* se llamará al evento **removeEventListener()**.

```
function acercar(e:MouseEvent):void {
      stage.addEventListener(Event.ENTER_FRAME,ciclo);
}

 //La función ciclo que incrementa en 10 la posición
 //del cuadrado y aumenta en 5 el tamaño del cuadrado
function ciclo(event:Event):void {
      mccua.y+=10;
      mccua.height+=5;
      mccua.width+=5;

 //Se pone la condición de que cuando llegue a 300
 //el evento ENTER_FRAME deje de ejecutarse
      if (mccua.y>310) {
            stage.removeEventListener(Event.ENTER_FRAME,ciclo);
      }
}
```

➢ CON VIDEO

En esta prueba se han realizado muy pocos cambios con respecto a la prueba anterior.

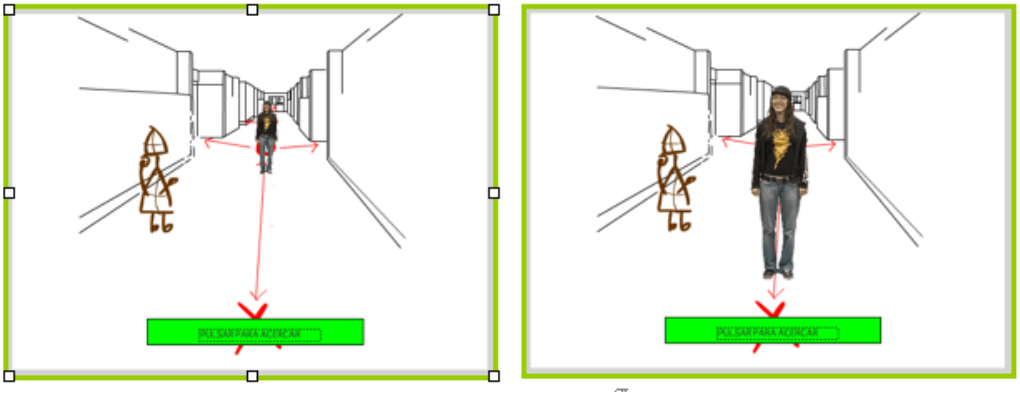


Figura 5.2.2 Estado inicial y final. Prueba crear perspectiva con vídeo.

El movieclip *mccua* de la prueba anterior, se ha sustituido por un video real ya editado al que se le ha llamado *carol.* El botón *pulsar* es el mismo que en la prueba anterior.

Se ha puesto un fondo detrás que simula un pasillo, así se podrá comprobar mejor el efecto de la perspectiva. La programación es exactamente la misma que en la prueba anterior.

Con esta prueba se ha comprobado simplemente, que a los vídeos, al igual que a los movieclips se les pueden cambiar las propiedades mediante ActionScript 3.0. Esto es muy útil, si se quiere agrandar, ocultar, mover un video, etc.

La prueba puede verse en **PespectivaVideo.fla**

5.2.2. Movimiento de objetos

Objetivo:

El círculo verde situado en el escenario se colocará en el lugar donde se haga click con el ratón.

Para ello únicamente se necesita un movieClip que se llamará *mcbola*. Se pulsará en cualquier parte del escenario y el objeto se colocará en el punto donde se hizo click.

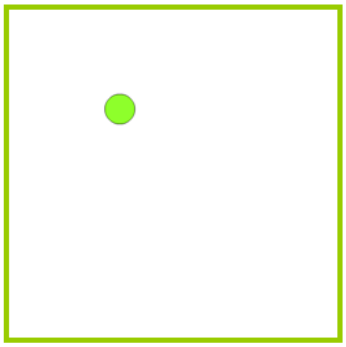


Figura 5.2.3 Estado inicial. Movimiento objetos

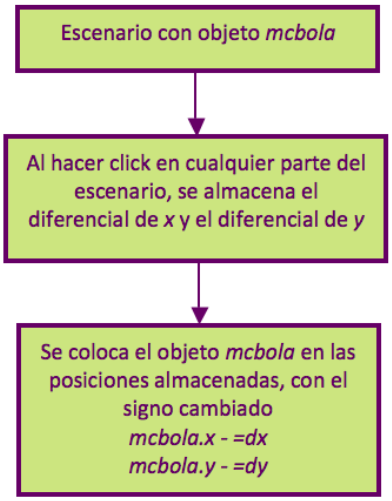


Diagrama 5.2. Movimiento de objetos

Código:

En primer lugar se crearán dos variables que se usarán para almacenar la posición donde se hizo click, es decir los diferenciales de *x* y de *y*.

```
var dx:int;
var dy:int;
```

Al hacer click en cualquier parte del escenario se detecta el evento y se llama a la función **muevebola()**.

```
stage.addEventListener(MouseEvent.CLICK,muevebola);
```

La función **muevebola()** almacena en las variables creadas, la distancia en *x* y en *y* respectivamente, que hay entre la posición del objeto *mccua* y el punto donde se hizo click, luego coloca el objeto en esa posición, cambiándole el signo.

```
function muevebola(event:MouseEvent):void {

 //Se almacena la distancia con número negativo
      dx=mcbola.x-mouseX;
      dy=mcbola.y-mouseY;

 //Se cambia el signo de la posición, para colocarlo en
 //el lugar correcto
      mcbola.x -= dx;
      mcbola.y -= dy;
}
```

➢ MOVER MOVIECLIP

Objetivo:

Presionando uno de los botones con forma de flecha, la bola azul se moverá en la dirección que indique la flecha.

Para esta prueba se necesitan tres movieClips , uno que será el objeto: *mcbola* y otros dos que serán las dos flechas: *izquierda* y *derecha*, que servirán para hacer mover el objeto hacia el lado correspondiente.

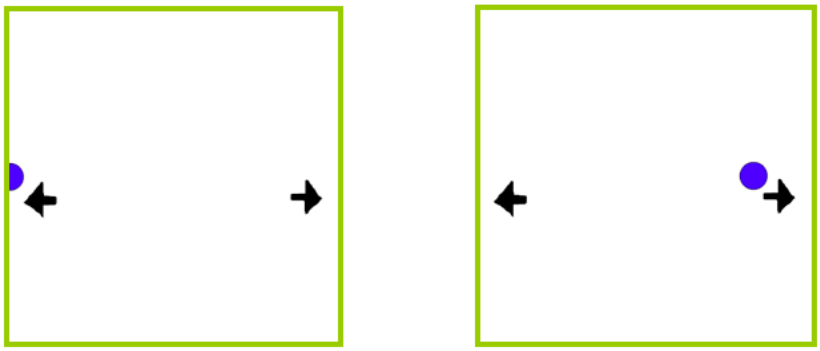


Figura 5.2.4 Estado inicial y final. Mover movieclip

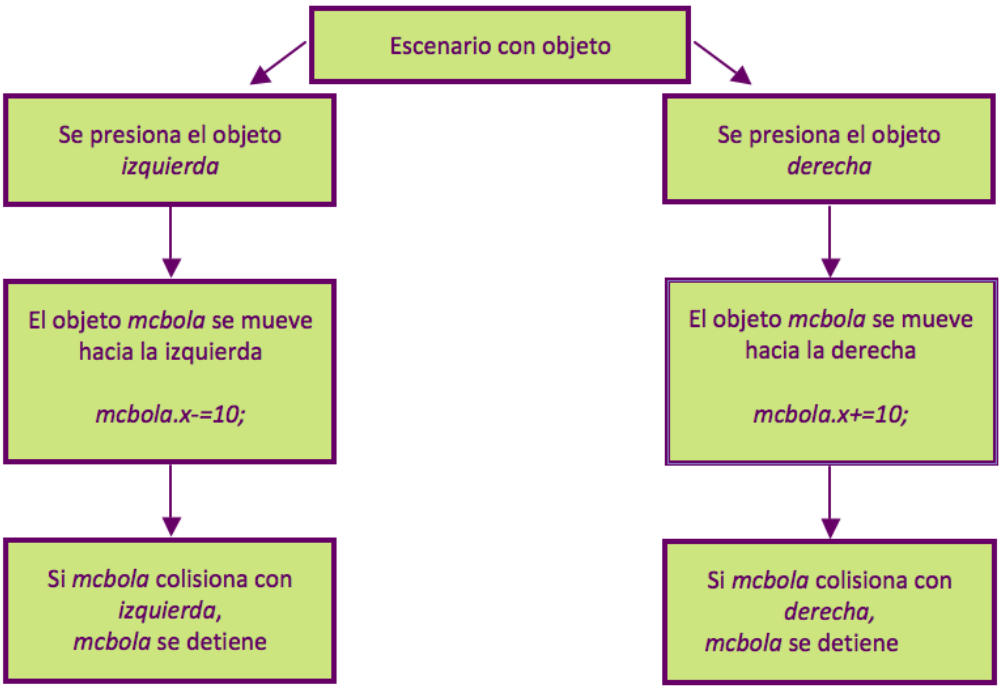


Diagrama 5.3. Mover Movieclip

Código:

Inicialmente se tiene el objeto en la posición:

```
mcbola.x=0;
mcbola.y=250;
```

Se crearán los detectores de evento de ratón para los objetos *izquierda* y *derecha*, que llevan asociados la función **moverizquierda()** y **moverderecha()** respectivamente.

```
derecha.addEventListener(MouseEvent.CLICK,moverderecha);
izquierda.addEventListener(MouseEvent.CLICK,moverizquierda);
```

Estas dos funciones llaman a un evento **ENTERFRAME**, que a su vez llama a las funciones **cicloderecha()** y **cicloizquierda()** respectivamente.

```
function moverderecha(e:MouseEvent):void {
       stage.addEventListener(Event.ENTER_FRAME,cicloderecha);
}

function moverizquierda(e:MouseEvent):void {
       stage.addEventListener(Event.ENTER_FRAME,cicloizquierda);
}
```

La función **cicloizquierda** hace que la posición de *mcbola* avance hacia la izquierda 10 píxeles cada vez, es decir va restando. Cuando *mcbola* colisiona con *izquierda* llama al evento **removeEventListener()**, por lo que *mcbola* se detiene.

```
function cicloizquierda(event:Event):void {
      mcbola.x-=10;
      if (mcbola.hitTestObject(izquierda)){

      stage.removeEventListener(Event.ENTER_FRAME,cicloizquierda);
      }
```

La función **cicloderecha** hace lo mismo que **cicloizquierda** solo que en vez de restar, suma 10 píxeles su posición y al colisionar con *derecha* se detiene.

```
function cicloderecha(event:Event):void {
      mcbola.x+=10;
      if (mcbola.hitTestObject(derecha)){

      stage.removeEventListener(Event.ENTER_FRAME,cicloderecha);
      }
```

➢ MOVER VIDEO

Esta prueba se realizará conjuntamente con la prueba siguiente.

5.2.3. Cambiar gestos de los vídeos

Las pruebas **mover vídeo** y **cambiar gestos de los vídeos**, tratan sobre la interactividad del vídeo, por lo que se decidió realizar una prueba que abarcara las dos. La interactividad vendrá marcada por el uso y control de los cuepoint, así que no habrá muchas diferencias a la hora de hacer que el personaje se mueva hacia un lugar, que al hacer que haga un determinado gesto, ya que todos los movimientos están editados uno detrás de otro en el mismo vídeo. Para acceder a cada uno, se utilizan los cuepoints, estos se colocarán al inicio y al final del movimiento.

Objetivo:

Al pulsar en un punto del escenario, el protagonista se dirigirá hacia ese punto, y al llegar realizará los gestos oportunos, para ello, primero se tendrá que valorar la posición actual del personaje y a continuación ejecutar las acciones que correspondan.

Para esta prueba el escenario dispondrá de siete movieclips en total, tres harán la función de partes activas (zonas con transparencia en las *figuras 5.2.5 y 5.2.6*), tres serán las marcas de posición (círculos en el suelo) y una hará de sombra del personaje.

Al pulsar sobre una de las partes activas, se evaluará la posición en la que se encuentra el personaje almacenándose en una variable. Esta evaluación se hace en el momento que la sombra coincide con una de las marcas de posición. Una vez se tiene la posición en la que se encuentra el personaje, se ejecutan las funciones que correspondan, accediendo a los cuepoints y moviendo la sombra hacia la posición correspondiente. Más adelante se dará un ejemplo más concreto.

El hecho de que se haya tenido que crear un movieclip que simula la sombra del personaje es porque aunque en el vídeo, solo se vea al personaje, (recordamos que el fondo es transparente), el vídeo abarca todo lo que corresponde al fondo que se tenía y ahora no se ve, por lo tanto si se quiere detectar que el vídeo se encuentra en una posición, no se puede calcular directamente con éste, ya que los límites del video están por delante y por detrás del personaje, esto se entiende mejor viendo la *figura 5.2.5.*

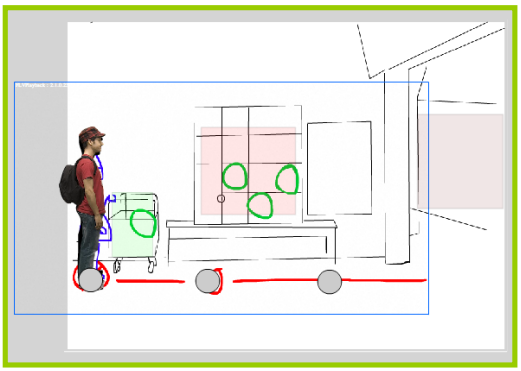


Figura 5.2.5. Video del personaje

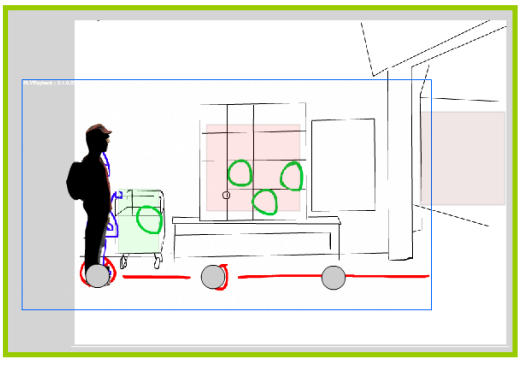


Figura 5.2.6 Sombra del personaje

Se ve que el vídeo ya esta por encima de las tres marcas de posición, por lo que se crea la sombra, *figura 5.2.6.*(encima del vídeo de la parte del personaje), para detectar la colisión con las marcas.

Este diagrama explica muy brevemente el funcionamiento:

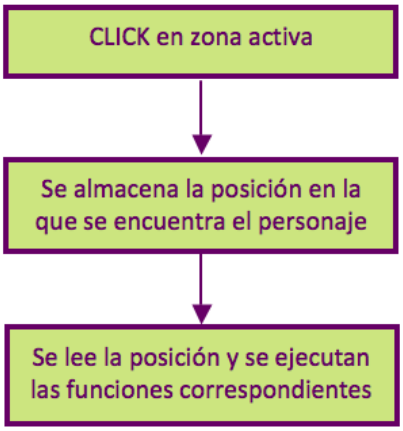


Diagrama 5.4. Cambiar gestos de los vídeos

Para entenderlo bien, se dará un ejemplo más completo, en concreto, cuando se hace click en la zona activa del *carrito.* En esta zona se hace un gesto de "coger algo". La posición que le corresponde a esta zona es la px, por lo que, el personaje si se encuentra en otra posición primero hará el gesto del movimiento de andar hacia px y cuando llegue, entonces hará el gesto de coger.

Se detallará cada función que se ejecuta dependiendo de donde se encuentre.

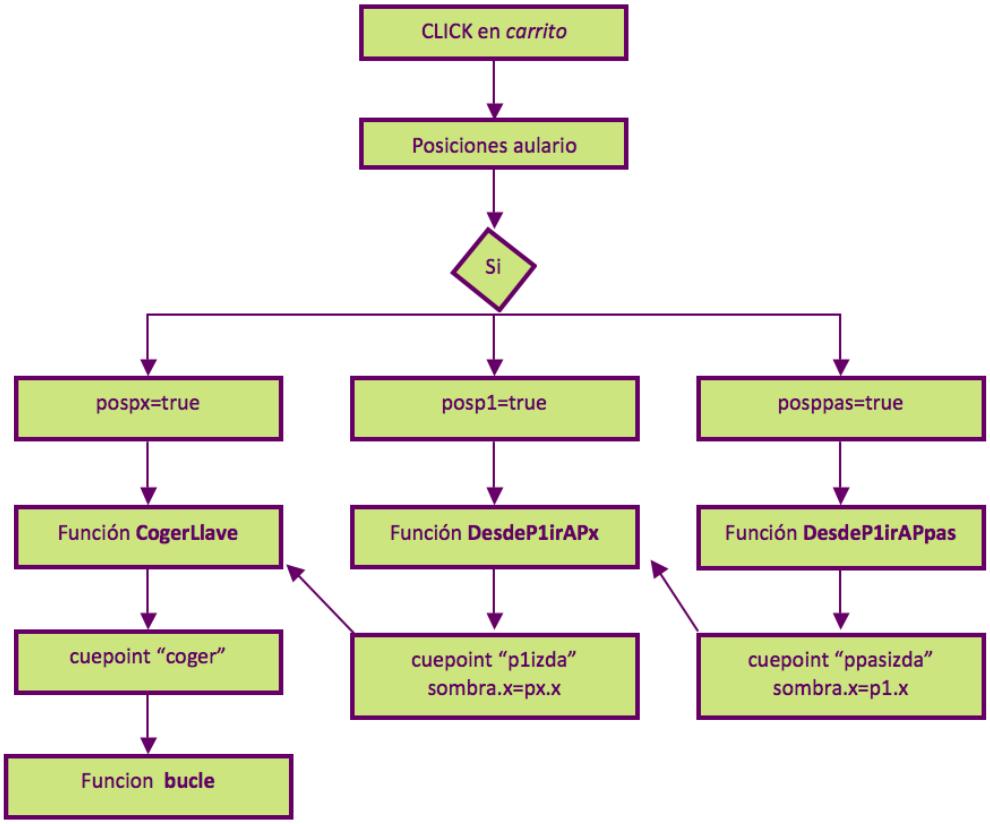


Diagrama 5.5. Cambiar gestos de los vídeos. Pulsar en zona activa carrito

Código:

Lo primero que se hará es importan la clases necesarias. A continuación, se crea la función **bucle**, que hace que el vídeo se quede haciendo bucle en un tipo de movimiento hasta que se pulsa en una de las zonas activas.

```
import fl.video.*;
import fl.video.MetadataEvent;
stop();

charli.addEventListener(MetadataEvent.CUE_POINT, bucle);
//Evento para que al principio el protagonista haga bucle

function bucle(eventObject:MetadataEvent):void {
      if(eventObject.info.name == "pxquietofin") {
            charli.seekToNavCuePoint("pxquieto");
      }
}
```

Después se crean las variables del tipo *booleano* que indicarán la posición en la que se encuentra la sombra y por lo tanto el vídeo.

Al principio se colocará la sombra en la misma posición que la marca *px* y se ejecuta la función **PosicionesAulario()**, esta función evalúa la posición de la sombra, comprobando si ha colisionado con una de las marcas y dependiendo en que marca se encuentre la variable correspondiente se pondrá en **true**.

```
//Variables que me dicen donde está el personaje

var pospx:Boolean;
var posp1:Boolean;
var pospasillo:Boolean;
sombra.x=px.x
PosicionesAulario();

//Posiciones aulario:
//Función que, según las variables, indica donde está el personaje

function PosicionesAulario(){

      if(sombra.x==px.x){
         pospx=true;
         posp1=false;
         pospasillo=false;
      }

      if(sombra.x==p1.x){
         pospx=false;
         posp1=true;
         pospasillo=false;
      }

      if(sombra.x==ppasillo.x){
         pospx=false;
         posp1=false;
         pospasillo=true;
      }
}
```

Ahora se analizará la parte en la que se hace click en una zona activa y el personaje ejecuta los movimientos correspondientes.

Esta en concreto, corresponde al *carrito* que coincide con la posición *px*, por lo que en esta posición terminarán todos lo movimientos.

Cuando se hace click en una zona activa lo primero se evalúa la posición con la función **PosicionesAulario()** y dependiendo donde se encuentre se ejecutará una función u otra.

```
//Al hacer click en CARRITO
carrito.addEventListener(MouseEvent.CLICK,irCarrito);

//////////////////  CARRITO  ////////////////////

function irCarrito(event:MouseEvent){

trace (pospx)
trace (posp1)
trace (pospasillo)
/*Dependiendo de la posición que se haya almacenado, se accede a una
función u otra*/
PosicionesAulario();

      if(pospx==true){
            CogerLlave();
      }

      if(posp1==true){
            DesdeP1IrAPx();
      }

      if(pospasillo==true){
            DesdePpasIrAP1();
      }
}
```

Si el personaje se encuentra en la posición *px*, se llamará a la función **CogerLlave()**, que sus acciones simplemente son, pasar a un cuepoint donde está el movimiento de coger y cuando termina el movimiento vuelve al movimiento del bucle.

```
function CogerLlave(){
//Movimiento coger la llave
      charli.seekToNavCuePoint("pxcoger");
      charli.addEventListener(MetadataEvent.CUE_POINT, QuedarseEnPx);
}
            function QuedarseEnPx(eventObject:MetadataEvent):void {
            /*Si se detecta el final del movimiento coger el video pasa
            al cuepoint para que haga bucle*/
            if(eventObject.info.name == "pxcogerfin") {
                        charli.seekToNavCuePoint("pxquieto");
                  }
            }
```

Si se encuentra en *p1*, se ejecuta la función **DesdeP1irAPx()**, esta función es más complicada. Primero pasa al cuepoint que indica el movimiento de andar hacia la izquierda que tendrá que hacer para llegar a la posición *px*, a la vez también se manda que la sombra ocupe el lugar px. Cuando se termina el movimiento de andar a la izquierda se hace que se ejecute la función **CogerLlave()** como se ha hecho en la primera parte.

```
function DesdeP1IrAPx(){
//Movimiento desde las taquillas a la posición px
      charli.seekToNavCuePoint("p1izda");
      charli.addEventListener(MetadataEvent.CUE_POINT,MovimientoCoger)
;
      //A la vez se mueve la sombra para detectar la posición
      sombra.x=px.x;
      PosicionesAulario();
}

            function MovimientoCoger(eventObject:MetadataEvent):void {
                  if(eventObject.info.name == "p1izdafin") {
                        CogerLlave();
                  }
```

La última parte se ejecuta si el personaje se encuentra en la posición *ppas*. Para esta función primero se hace el movimiento de *ppas* a *p1* y a continuación se ejecuta la función que se ha explicado antes, así, de esa forma, el personaje pasa desde *ppas* a *p1* y luego de *p1* a *px*.

```
//Movimiento desde pasillo a las taquillas
function DesdePpasIrAP1(){

      charli.seekToNavCuePoint("ppasizda ");
      charli.addEventListener(MetadataEvent.CUE_POINT,MovimientoDesdeP
pasAP1);
      sombra.x=p1.x;
      PosicionesAulario();
}
/*Detecta que llega a las taquillas y pasa a la función de ir hasta el
carrito*/

function MovimientoDesdePpasAP1(eventObject:MetadataEvent): void {
                  if(eventObject.info.name == "ppasizdafin") {
                        DesdeP1IrAPx();
                  }
}
```

Se comprobó que estas pruebas daban en determinados momentos errores, por ejemplo, si se pulsaba en la zona activa de las *taquillas,* estando en la posición *px*, el personaje en vez de avanzar hasta la posición *p1* y detenerse en ese punto, avanzaba hasta *ppas*. Esto ocurría si antes se había pulsado en la zona activa del *pasillo*.

Para solucionar esto se reescribió otra prueba para ver con más detalle el funcionamiento de los cuepoints. La prueba es la siguiente:

➢ FUNCIONAMIENTO DE LOS CUEPOINTS

Se editó un vídeo formado por varios fondos de diferentes colores (morado azul, verde, blanco rojo y negro) colocados cada 5 segundos. Al inicio del fondo de color se colocó un cuepoint con el nombre del color correspondiente ("amarillo") y al final, se colocó otro para indicar que había terminado ese color ("finamarillo").

En flash se colocaron 6 botones rectangulares de los colores de los fondos de los vídeos, al pulsar en uno de los botones, el vídeo se accede al cuepoint que indica el color y por lo tanto se ve el fondo correspondiente.

A parte hay dos botones redondos, uno verde oscuro y otro morado. La función de estos dos botones es:

- Al pulsar el botón verde el vídeo pasa por el fondo amarillo, después por el azul y para terminar por el verde.

- Al pulsar el botón morado pasa por el fondo azul, luego al rojo y para terminar al morado.

Es decir los dos botones tienen en común el cuepoint "azul" y "finazul", pero cada uno manda visualizar un color diferente.

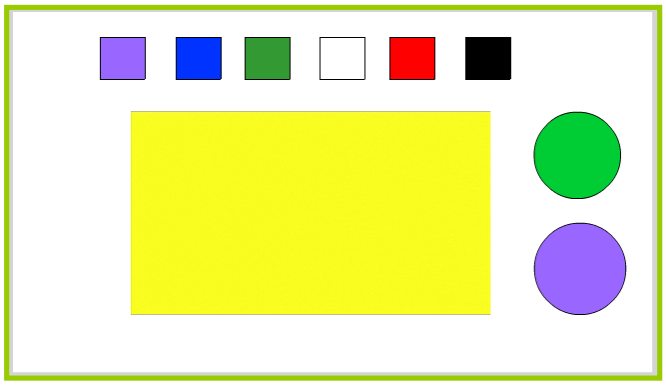


Figura 5.2.7 Estado inicial y final. Prueba crear perspectiva

Lo que ocurre es que, si se pulsa cualquiera de los dos botones primero lo hace correctamente, pero si después se pulsa el otro botón, lo que ocurre es que primero hace lo correspondiente al primer botón y luego al que se ha pulsado, es decir, se almacena lo que ya hizo cuando detecta un cuepoint y cada vez que pasa por él, aunque se ejecute otra función, vuelve a ejecutar lo primero que hizo. Esto se puede ver más de forma más sencilla, con la aplicación **cue.fla.**

Para terminar, se realizó una última prueba sobre vídeo. Es igual que la prueba 5.2.3 solo que en esta, se colocó un cuepoint para cada acción, de forma que no se repitiera ninguno. Los resultados fueron los esperados, el personaje no realizaba los gestos correctos en cada momento. Además se añadió la función que permitía que, al pulsar la zona activa *pasillo*, el personaje en vez de quedarse en la posición *ppas*, avanzaba hasta el siguiente escenario, que es lo que tendrá que ocurrir en la aventura gráfica. Esta prueba corresponde con **posiciones.fla.**

5.2.4 Mover el escenario.

Objetivo:

Presionando uno de los dos botones con forma de flecha, se avanzará por el escenario hacia el lado que indique la flecha.

Para ello se necesitan cuatro movieClips, el escenario propiamente dicho, llamado *escenario*, los dos movieClips, con forma de flecha que actuarán de botones: *flechaizda* y *flechaderecha* y el objeto *limite,* que hace que el escenario se detenga en el límite de la parte izquierda. Al hacer click en el objeto *flechaderecha*, el escenario se moverá hacia la izquierda, dejando ver la parte derecha, al llegar a la posición *x=-500*, el *escenario* se detendrá y al revés al pulsar la *flechaizda,* este se detendrá al colisionar con *límite.*

Figura 5.2.8 Estado inicial y final. . Mover escenario

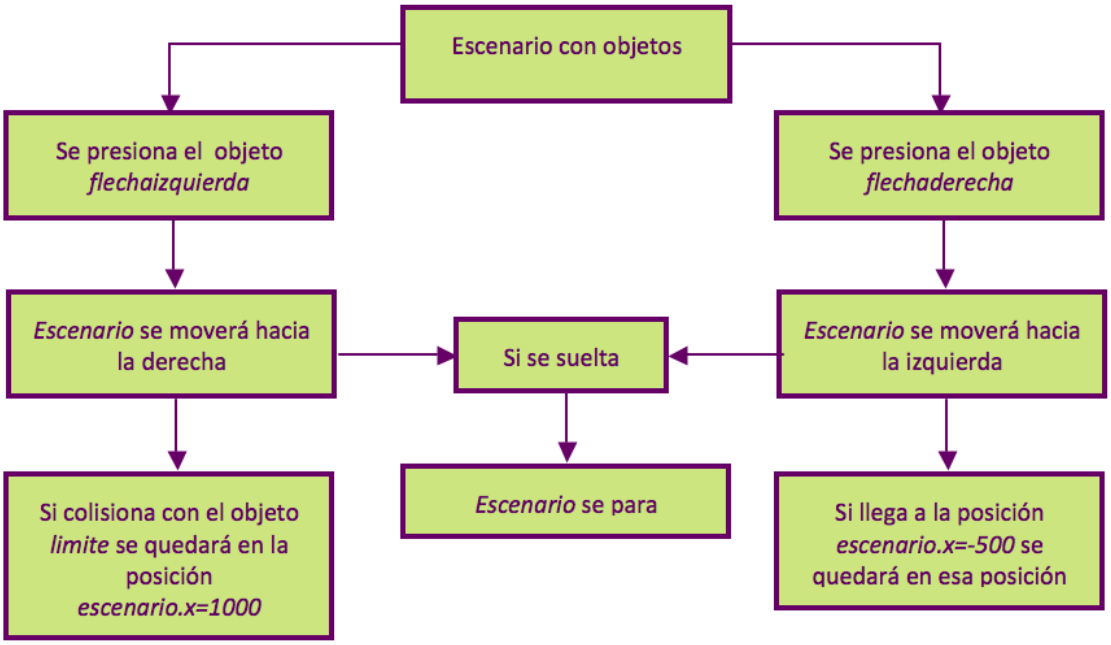


Diagrama 5.6. Mover escenario

Código:

Inicialmente se tiene el *escenario* en la posición *x=1000* además se ha colocado un objeto *limite* que hará de límite para la parte izquierda.

```
escenario.x=1000;
escenario.y=250;
 //Posición de la barra de límite de escenario.
limite.x=2022;
```

Se crearán dos detectores de eventos para las flechas, asociadas a las funciones **moverizquierda()** y **moverderecha()** respectivamente, estas funciones llamarán a un evento **ENTERFRAME** que llama a las funciones **cicloizda()** y **cicloderecha()**.

También se tienen unos detectores de eventos en *flechaizda* y *flechaderecha* para que el *escenario* se detenga.

```
flechaizda.addEventListener(MouseEvent.MOUSE_DOWN,moverizda);
flechaizda.addEventListener(MouseEvent.MOUSE_UP,parar);

flechaderecha.addEventListener(MouseEvent.MOUSE_DOWN,moverderecha);
flechaderecha.addEventListener(MouseEvent.MOUSE_UP,parar);

 //Para ver la parte derecha del escenario se llamará al
 //evento ENTER_FRAME, que lleva la función cicloderecha asociada

function moverderecha(e:MouseEvent):void {
      stage.addEventListener(Event.ENTER_FRAME,cicloderecha);

}

 //Para ver la parte izquierda del escenario se llamará al
 //evento ENTER_FRAME, que lleva la función cicloizquierda asociada

function moverizda(e:MouseEvent):void {
      stage.addEventListener(Event.ENTER_FRAME,cicloizda);

}
 //La función parar se activa cuando se suelta el ratón
function parar(e:MouseEvent):void {
      stage.removeEventListener(Event.ENTER_FRAME,cicloderecha);
      stage.removeEventListener(Event.ENTER_FRAME,cicloizda);
}
```

La función **cicloizda()** dejará ver la parte izquierda del escenario, por lo que se desplazará hacia la derecha, sumando 10 píxeles cada vez. Cuando el *escenario* colisione con el objeto *limite*, se detendrá en la posición *x = 1000* que es el límite por la parte izquierda.

```
function cicloizda(event:Event):void {
      escenario.x+=10;
      if(escenario.hitTestObject(limite) ){
            escenario.x=1000;
      }
}
```

La función **cicloderecha()** hará lo contrario y cuando el escenario llegue a la posición en *x = -500*, se llamará al evento **removeEventListener()**.

```
function cicloderecha(event:Event):void {
      escenario.x-=10;
      if (escenario.x < -500) {
            escenario.x=-499;
      }
}
```

5.2.5. Crear lugar para almacenar los objetos.

Objetivo:

Al presionar los objetos situados en el escenario, el cuadrado rojo o el círculo azul, estos se colocarán en la parte inferior del escenario, donde se tendrá el inventario.

Para ello se tendrá el escenario, donde se colocarán dos movieClips llamados *objeto* y *objeto2*, que se meterán en un contenedor llamado *objetos*. Se creará un contenedor llamado *inventario* donde se meterán los objetos ocultos con nombres *objetoin* y *objeto2in*. Al hacer click en uno de los objetos del escenario, se ocultará el y se pondrá en visible el correspondiente del inventario.

Figura 5.2.9 Estado inicial y final. Crear lugar para almacenar objetos

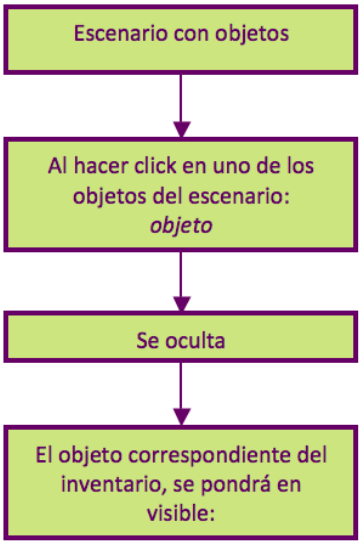


Diagrama 5.7. Crear lugar para almacenar objetos

Código:

Lo primero que se hace es crear un contenedor para los objetos del escenario y para los objetos del inventario, llamados *objetos* e *inventario.*

```
  //Contenedor para los objetos del escenario
var objetos:Sprite=new Sprite();
this.addChild(objetos);

  //Se meten los objetos al contenedor
objetos.addChild(objeto2);
objetos.addChild(objeto)

 //Se crea otro contenedor que será el inventario y los objetos del
 //inventario
var inventario:Sprite=new Sprite();
this.addChild(inventario);

inventario.addChild(objeto2in);
inventario.addChild(objetoin);
```

Se ocultan los objetos del inventario.

```
objeto2in.visible=false;
objetoin.visible=false;
```

Se crea un detector de eventos de ratón para el contenedor *objetos*, que llama la función **coger().** Esta función hace que al hacer click sobre un objeto del escenario este se vuelva invisible y aparezca su correspondiente en el inventario

```
objetos.addEventListener(MouseEvent.CLICK, coger);

 function coger(Event:MouseEvent):void{
      Event.target.visible=false;
      inventario.getChildByName(Event.target.name +
"in").visible=true;
 }
```

5.2.6. Coger objetos y arrastrar.

Objetivo:

Presionando el cuadrado azul se podrá arrastrar con el ratón, hasta que se suelte este.

Para ello se necesita un movieClip llamado *mcuadrado* que en el momento que se haga click, se podrá arrastrar el objeto y cuando se suelte, el objeto se quedará en el lugar que se suelte el ratón.

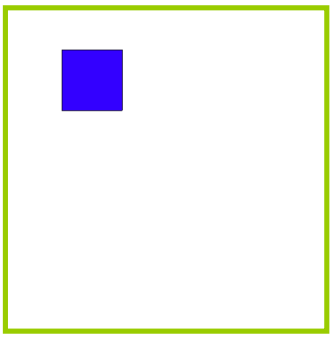


Figura 5.2.10 Estado inicial. Coger objeto y arrastrar

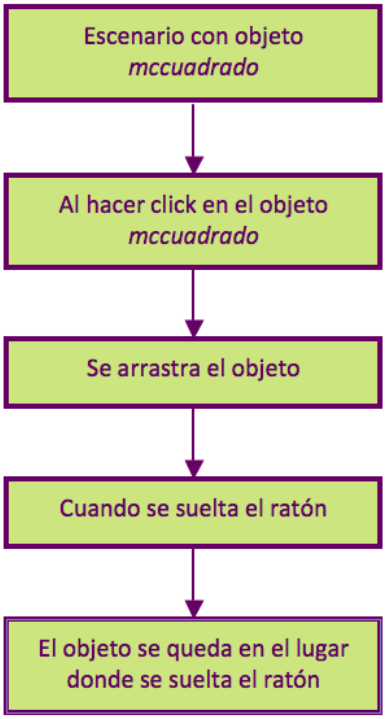


Diagrama 5.8. Coger objeto y arrastrar

Código:

Se crean dos detectores de eventos para el objeto, cuando se pulse el ratón y cuando se suelte, llevaran asociadas las funciones **arrastrar()** y **noarrastrar()** respectivamente.

```
mccuadrado.addEventListener(MouseEvent.MOUSE_DOWN, arrastrar);
mccuadrado.addEventListener(MouseEvent.MOUSE_UP, noarrastrar);
```

Con la función **arrastrar()** se llamará al método **startdrag();**

```
function arrastrar(Event:MouseEvent):void{
      mccuadrado.startDrag(); //Empezar a arrastrar con el ratón
}
```

Con la función **noarrastrar()** se llamará al método **stopdrag()**

```
function noarrastrar(Event:MouseEvent):void{
      mccuadrado.stopDrag(); //Parar de arrastrar
 }
```

5.2.7. Coger objetos, arrastrar y colocar

Objetivo:

Presionando en cada rectángulo de color, se tendrá que arrastrar y colocar en sus respectivos rectángulos grises, donde viene el nombre del color de cada rectángulo. Si no se coloca en el lugar que le corresponde volverán a su posición inicial.

Para ello se necesitarán tres movieClips que serán los objetos *mcamarillo, mcazul* y *mcmorado*, se colocarán en un contenedor *objetos*. Se tendrán seis movieClips más, tres de ellos no se verán, que harán de guía para la posición inicial de los objetos llamados *mcamarilloinicial, mcazulinicial* y *mcmoradoinicial* y tres movieClips que harán de guía para la posición final, llamados *mcamarillopos, mcazulpos* y *mcmoradopos.* Al arrastrar el objeto hasta su casilla correspondiente, cuando se suelte el ratón, si es la correcta se quedara dentro de la casilla, si no es, volverá a su posición inicial.

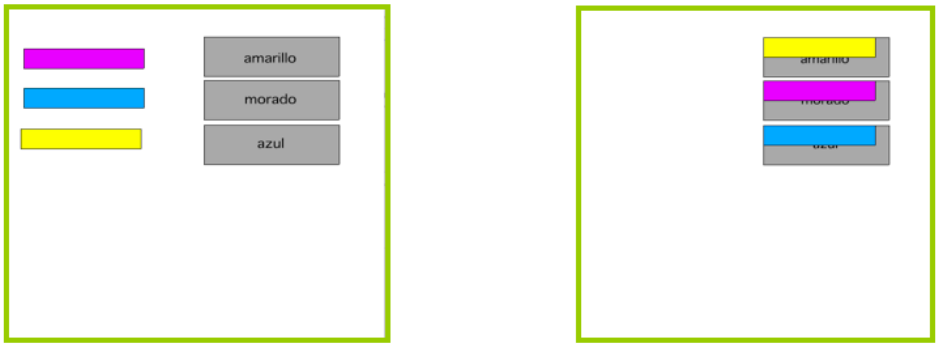


Figura 5.2.11. Estado inicial y final. Coger objeto, arrastrar y colocar

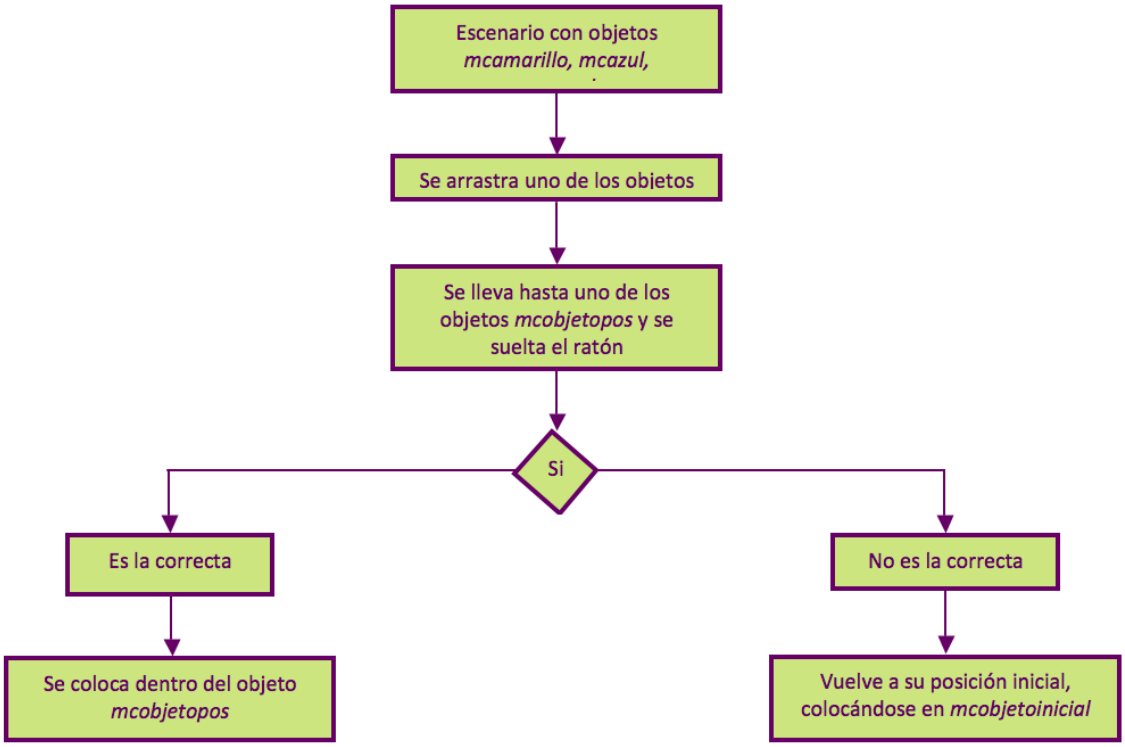


Diagrama 5.9. Coger objeto, arrastrar y colocar

Código:

Se crea el contenedor *objetos* para los objetos que se mueven.

```
var objetos:Sprite=new Sprite();

this.addChild(objetos);
objetos.addChild(mcmorado);
objetos.addChild(mcazul);
objetos.addChild(mcamarillo);
```

Se añaden los detectores de eventos al contenedor, para arrastrar los objetos y para parar de arrastrar.

```
objetos.addEventListener(MouseEvent.MOUSE_DOWN, arrastrar);
objetos.addEventListener(MouseEvent.MOUSE_UP, noarrastrar);
```

Al hacer click sobre uno de los objetos, se comienza a arrastrar. Con la propiedad target se hace referencia al objeto elegido dentro del contenedor.

```
function arrastrar(Event:MouseEvent):void{
      Event.target.startDrag();
}
```

Cuando se suelta el ratón se llama a la función **noarrastrar()**, en la que pueden ocurrir dos cosas: si el *mcobjeto* choca con su correspondiente, se colocará dentro del *mcobjetopos*. Sino volverá a su posición inicial, que es dentro de *mcobjetoinicial*, que no se verá.

```
function noarrastrar(Event:MouseEvent):void {

if (Event.target.hitTestObject(getChildByName(Event.target.name + "pos"
            Event.target.x =getChildByName(Event.target.name + "pos").x
            Event.target.y =getChildByName(Event.target.name + "pos").y

            Event.target.stopDrag();
 }

 //Sino volverá a su posición inicial, que es dentro de un
 //rectángulo blanco.
else  {
            Event.target.x =getChildByName(Event.target.name +
"inicial").x;
            Event.target.y =getChildByName(Event.target.name +
"inicial").y;

            Event.target.stopDrag();
```

5.2.8. Combinar objetos.

Objetivo:

Al Arrastrar el círculo rojo sobre el cuadrado azul o viceversa y soltando el ratón, estos dos objetos se ocultarán y aparecerá un triangulo verde.

Para ello se necesitan tres movieClips *mccuadrado*, *mccirculo* y *mctriangulo* que se colocarán en un contenedor llamado *objetos*. Al llevar un objeto hasta otro, este se colocará en la capa superior y al soltarlo encima del otro objeto, aparecerá el objeto *mctriangulo*

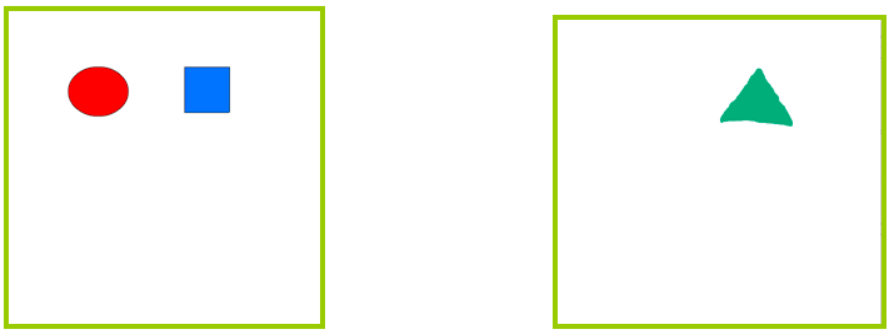


Figura 5.2.12. Estado inicial y final. Combinar objetos

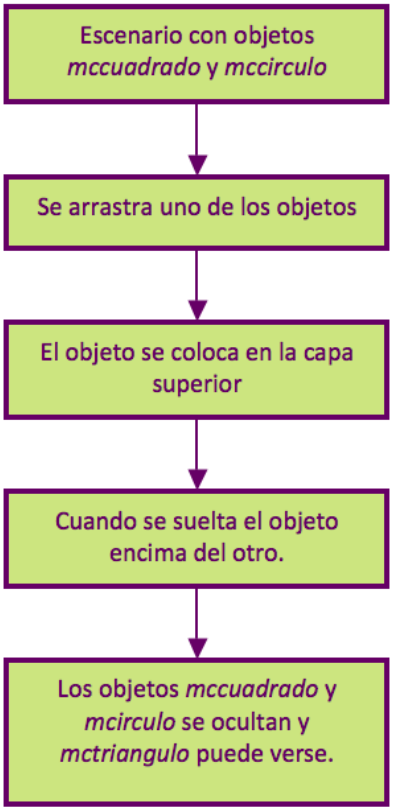


Diagrama 5.10. Combinar objetos

Código:

Se crea un contenedor que contendrá a los 3 objetos.

```
var objetos:Sprite=new Sprite();
this.addChild(objetos);

 objetos.addChild(mctriangulo);
 objetos.addChild(mccuadrado);
 objetos.addChild(mccirculo);
```

En un principio el *mctriangulo* estará oculto.

```
mctrianqulo.visible=false;
```

Se colocan los detectores de eventos en el contenedor, con las funciones **arrastrar()** y **parar()** asociadas.

```
 objetos.addEventListener(MouseEvent.MOUSE_DOWN,arrastrar);
 objetos.addEventListener(MouseEvent.MOUSE_UP,parar);
```

La función **arrastrar()** coloca el objeto que se hace click, en la capa más alta.

```
function arrastrar(Event:MouseEvent):void{
      Event.target.startDrag();
      if(Event.target==mccuadrado){
            contenedor.addChild(mccuadrado);
      }
      else{
            contenedor.addChild(mccirculo);
      }
 }
```

Al soltar el ratón, la función **parar()** hace que evalúe si ha colisionado con el otro objeto. Si ha colisionado, lo que ocurre es que los dos objetos se ocultan y aparece el objeto *mctriangulo* en la posición del objeto que se hace click.

```
function parar(Event:MouseEvent):void{
      Event.target.stopDrag()
      if(mccirculo.hitTestObject(mccuadrado)){

      mctriangulo.x=Event.target.x;
      mctriangulo.y=Event.target.y;
      mccirculo.visible=false;
      mccuadrado.visible=false;
      mctriangulo.visible=true;
      }
 }
```

5.2.9. Cambiar el escenario

Objetivo:

Presionando el botón situado en el escenario, se cambiará el fondo del escenario, los objetos que estén en cada escenario también se cambiarán.

Para ello se necesitan dos fondos llamados *fondonormal* y *fondoalien* y un movieClip que hará de botón, llamado *cambiar*. Se meterá cada escenario en un contenedor, *normal* y *alien* respectivamente, en contenedor *alien* se meterá un objeto llamado *cuadrado*. Al hacer click en el botón, se intercambiaran las capas de los escenarios y así se verá uno cada vez, el objeto *cuadrado* solo se verá cuando se vea *fondoalien.*

Figura 5.2.13 Estado inicial y final. Cambiar el escenario

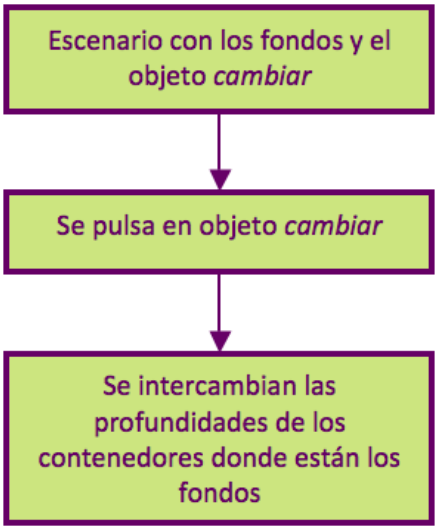


Diagrama 5.11. Cambiar el escenario

Código:

Se crean dos contenedores, *normal* y *alien*, donde se tendrán los escenarios.

```
var normal:Sprite=new Sprite();
this.addChild(normal);
normal.addChild(fondonormal);

var alien:Sprite=new Sprite();
this.addChild(alien);
alien.addChild(fondoalien);
alien.addChild(cuadrado);
```

Se crea un escuchador en el botón *cambiar* cuando se haga click, que llamará a la función **cambiarescenario()**. Esa función cambiará la profundidad de los contenedores.

```
cambiar.addEventListener(MouseEvent.CLICK,cambiarescenario);

function cambiarescenario (Event:MouseEvent):void {
      swapChildren (fondoalien,fondonormal)
}
```

5.2.10. Aparecer texto

Objetivo:

Al pasar el puntero del ratón por encima de cada objeto, aparecerá el nombre y el color del objeto sobre el que se encuentre.

Para ello se necesitan tres movieClips que serán los objetos, llamados *mctriangulo, mccuadrado,* y *mccirculo.* Se creará un área de texto y se colocará en el puntero del ratón oculta, cuando se pase por encima de uno de los objetos se volverá visible y aparecerá el texto correspondiente.

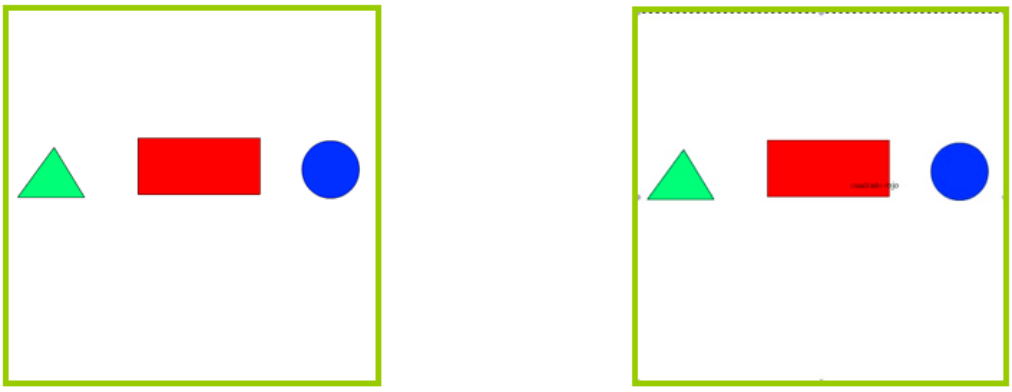


Figura 5.2.14. Estado inicial y final. Aparecer texto

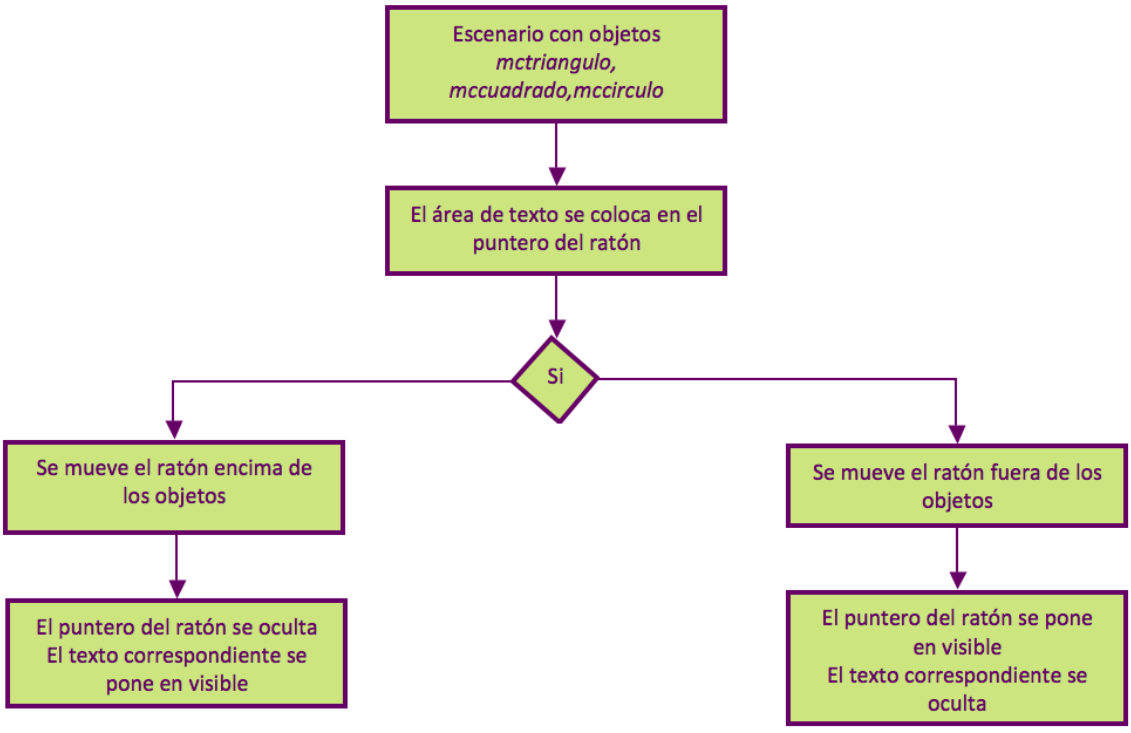


Diagrama 5.12. Aparecer texto

Código:

Se crea un campo de texto nuevo y se añade al escenario.

```
var texto:TextField = new TextField();
addChild(texto);
```

Se hace que el campo de texto se adapte al propio texto y que esté oculto.

```
texto.autoSize=TextFieldAutoSize.LEFT;
texto.visible=false;
```

Se añade un escuchador **ENTERFRAME** asociado a al función **movertexto().** Lo que ocurrirá al activarse el escuchador, es que el campo de texto creado, se colocará a 14 píxeles en *x* y en *y* del puntero del ratón. Pero no se verá porque estará oculto.

```
stage.addEventListener(Event.ENTER_FRAME,movertexto);

function movertexto(event:Event):void{
      texto.x=stage.mouseX+14;
      texto.y=stage.mouseY+14
}
```

Se asigna a cada objeto un texto diferente.

```
mccuadrado.text= "cuadrado rojo" ;
mctriangulo.text= "triangulo verde" ;
mccirculo.text= "circulo azul" ;
```

Se añade un escuchador al escenario, donde están los objetos para que cada vez que se pase el ratón por encima de ellos llame a la función **mostrartexto()** y otro cuando este fuera del objeto, que llamará a **ocultartexto().**

```
stage.addEventListener(MouseEvent.MOUSE_OVER,mostrartexto);

stage.addEventListener(MouseEvent.MOUSE_OUT,ocultartexto);
```

Al pasar por encima de los objetos, se asigna al campo de texto, el texto del objeto y cuando este fuera el campo de texto se ocultará.

```
function mostrartexto(Event:MouseEvent):void{
      texto.text=Event.target.text;
      texto.visible=true;
}
function ocultartexto(Event:MouseEvent):void{
      texto.visible=false
}
```

5.2.11. Condiciones para que ocurran cosas

Objetivo:

Si se presiona sobre una llave colocada en el escenario y a continuación se presiona sobre la puerta, aparecerá un texto, si no se ha presionado sobre la llave, al presionar la puerta, aparecerá otro texto diferente.

Para ello se colocará un movieClip, que será el objeto *llave*, en el escenario. Se tendrá un movieClip con alpha cero, *puerta*, en la puerta para poder interactuar con ella. Y habrá otros dos movieClips ocultos con el texto en el escenario, hasta que se pulse en la puerta, llamados *textsi* y *textno*. Si se pulsa la *llave*, se posicionará en un objeto llamado *objetocon* dentro del inventario, al hacer click en la *puerta* aparecerá *textsi* y si no se pulsa la *llave* al hacer click en la puerta nos aparecerá *textno.*

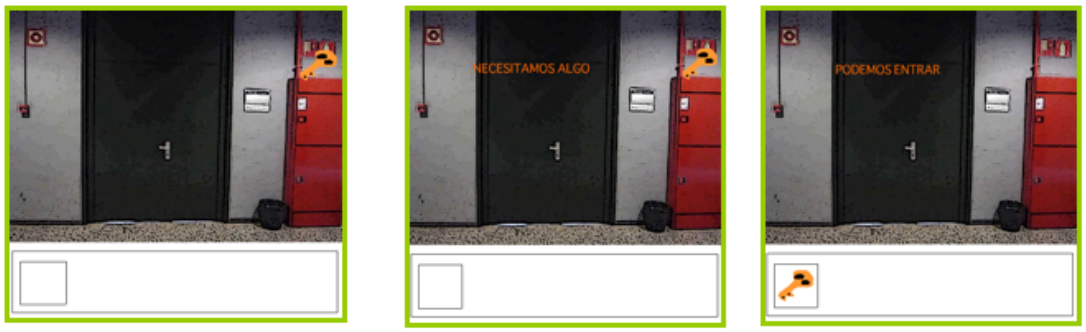


Figura 5.2.15. Estados. Condiciones para que ocurran cosas

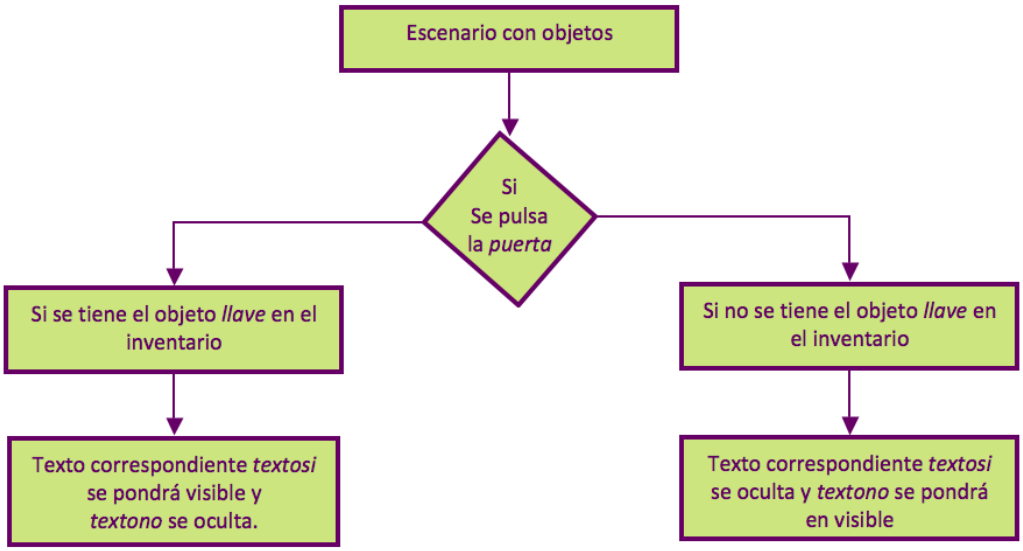


Diagrama 5.13. Condiciones para que ocurran cosas

Código:

Se ponen las condiciones iniciales de los objetos de la escena

```
puerta.alpha=0;
textsi.visible=false;
textno.visible=false;
```

La primera parte es la de coger la llave. Se crea un detector para la llave cuando se pulse, la función **muevellave()** se ejecuta y la llave se coloca en la posición del objeto, *llavecon*.

```
llave.addEventListener(MouseEvent.CLICK,muevellave);

function muevellave(event:MouseEvent):void {
      llave.x=llavecon.x
      llave.y=llavecon.y;
      textno.visible=false;
}
```

En la *puerta* se coloca un detector, que tiene asociada la función **comprobarpuerta()**, cuando se haga click sobre ella, si se tiene la *llave* en el inventario, aparecerá un texto y si no aparecerá otro.

```
puerta.addEventListener(MouseEvent.CLICK,comprobarpuerta)

function comprobarpuerta(Event:MouseEvent):void{
      if(llave.x == llavecon.x){
            textsi.visible=true
            textno.visible=false
            textsi.x=200;
            textsi.y=100;
      }
      else{
            textno.visible=true
            textsi.visible=false
            textno.x=200;
            textno.y=100;
      }
}
```

5.2.12. Cambiar el cursor

Objetivo:

Al pasar el puntero del ratón por encima de cada objeto, aparecerá un puntero diferente, dependiendo del objeto sobre el que se esté.

Para ello se necesitan tres movieClips, que serán los objetos, llamados *mcrectangulo, mccuadrado y mccirculo* y tres movieClips que representarán los punteros, *mcojo, mcboca* y *mcmano*. Los punteros se colocarán en la posición del puntero al comenzar la película pero estarán ocultos, al pasar por un objeto se verá el puntero correspondiente.

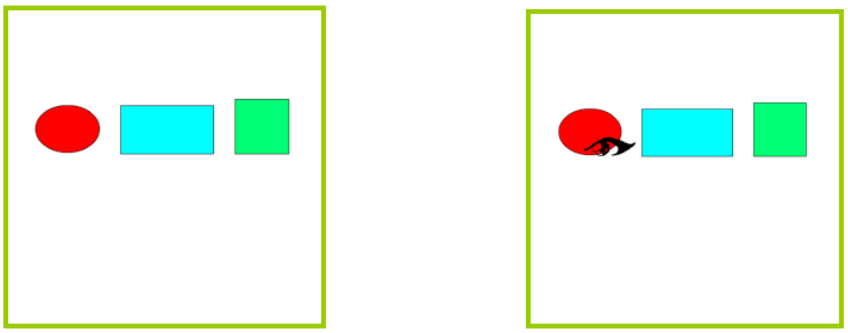


Figura 5.2.16. Estado inicial y final. Cambiar el cursor

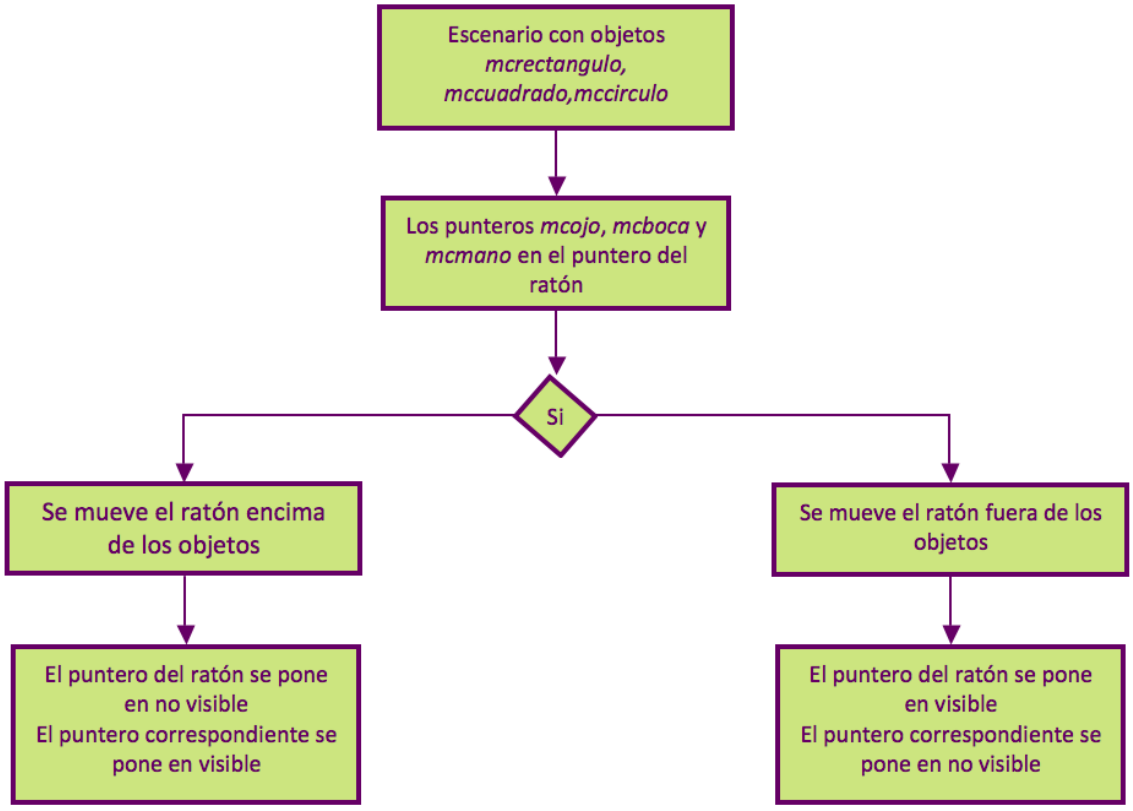


Diagrama 5.14. Cambiar el cursor

Código:

Se ocultan los tres movieclips que actuarán como puntero de ratón.

```
mcojo.visible=false;
mcmano.visible=false;
mcboca.visible=false;
```

Se crea un evento **ENTERFRAME** en el escenario que lleve asociado un función llamada **puntero().** Esta función hace que nada mas iniciar la película, los punteros se sitúen en la posición del puntero del ratón.

```
stage.addEventListener(Event.ENTER_FRAME,puntero)

function puntero (event:Event):void{
	mcojo.x=stage.mouseX;
	mcojo.y=stage.mouseY;
	mcmano.x=stage.mouseX;
	mcmano.y=stage.mouseY;
	mcboca.x=stage.mouseX;
	mcboca.y=stage.mouseY;
}
```

Se crean los escuchadores para cada objeto, cuando se pasa el ratón por encima, que llamarán a las funciones **mostrarojo()**, **mostrarmano()** y **mostrarboca(),** que ocultarán el puntero del ratón y dejarán ver el movieClip correspondiente que hará de nuevo puntero.

```
mccirculo.addEventListener(MouseEvent.MOUSE_OVER,mostrarojo);
mcrectangulo.addEventListener(MouseEvent.MOUSE_OVER,mostramano);
mccua


drado.addEventListener(MouseEvent.MOUSE_OVER,mostrarboca);

function mostrarojo(event:MouseEvent):void {
	Mouse.hide();

	mcojo.visible=true;
	mcojo.startDrag();
}

function mostramano(event:MouseEvent):void{
	Mouse.hide();

	mcmano.visible=true;
	mcmano.startDrag();
}

function mostrarboca(event:MouseEvent):void{
	Mouse.hide();

	mcboca.visible=true;
	mcboca.startDrag();
}
```

Cuando el ratón está fuera del objeto. Las funciones **ocultarojo()**, **ocultarmano()** y **ocultarboca()**, ocultarán los movieclips y dejarán ver el cursor normal del ratón.

```
mccirculo.addEventListener(MouseEvent.MOUSE_OUT,ocultarojo);
mcrectangulo.addEventListener(MouseEvent.MOUSE_OUT,ocultarmano);
mccuadrado.addEventListener(MouseEvent.MOUSE_OUT,ocultarboca);

function ocultarojo(event:MouseEvent):void{
      Mouse.show();

      mcojo.visible=false;
}
function ocultarboca(event:MouseEvent):void{
      Mouse.show();

      mcboca.visible=false;
}
function ocultarmano(event:MouseEvent):void{
      Mouse.show();

      mcmano.visible=false;
}
```

5.2.13. Introducir texto.

Objetivo:

Al escribir cualquier cosa en el área de texto y presionar el botón donde pone comprobar, aparecerá una cruz roja, para eliminarla se deberá presionar el botón donde pone volver a intentar. Si se escribe la palabra "irene" y se presiona el botón comprobar, aparecerá el tic verde.

Para ello se necesitan dos movieClips que harán de botones, uno para comprobar que se ha escrito lo correcto, *btn_comprobar* y otro para volver a intentar si no se ha hecho, *btn_volintentar*, dos movieClips, una cruz roja, *mal* o un tic verde, *ok*, que mostrará visualmente si se ha escrito lo correcto o no. Si se ha hecho bien aparecerá *ok*, y si no, *mal*. Al pulsar el objeto *btn_volintentar,* se ocultará el objeto *mal* y el área de texto se pondrá en blanco.

Figura 5.2.17. Estado inicial y final. Introducir texto

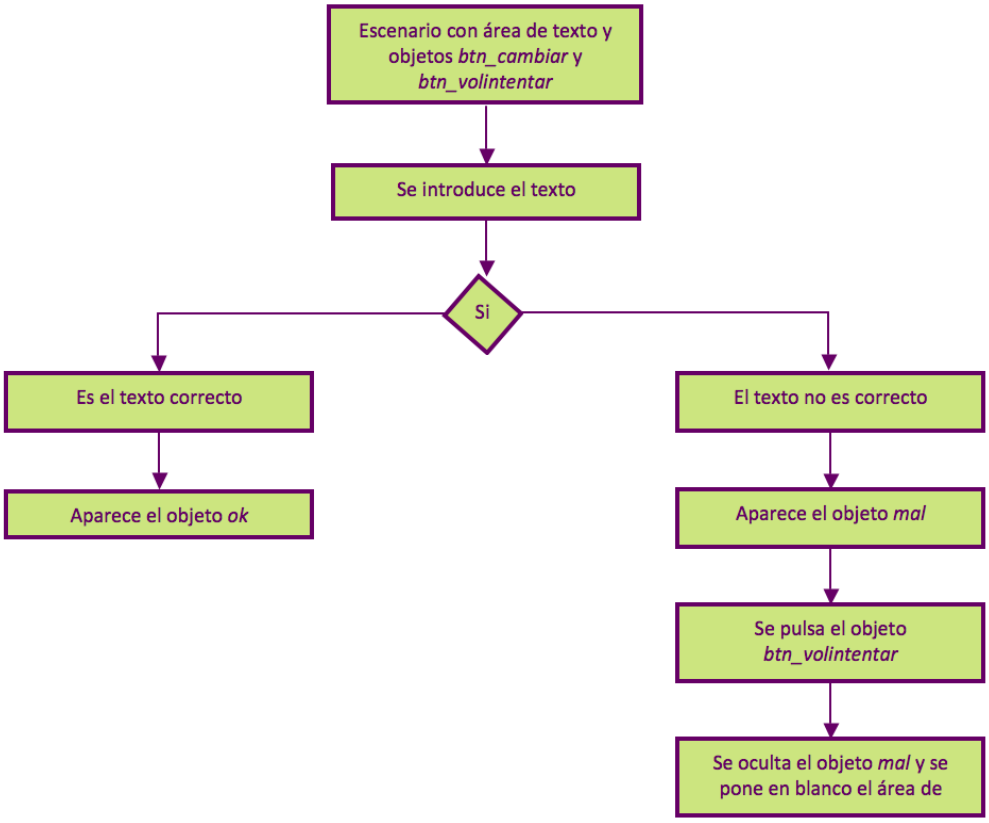


Diagrama 5.15. Introducir texto

Código:

Se colocan los objetos *mal* y *ok*, por defecto ocultos.

```
ok.visible=false;
mal.visible=false;
```

Se crea un área de texto, se posiciona y se le da un tamaño.

```
var nombre:TextArea = new TextArea();
 //La posicionamos y le damos un tamaño
nombre.move(200,80);
nombre.setSize(50,30);
 //La añadimos al escenario.
addChild(nombre);
```

El detector de eventos en el objeto *btn_comprobar,* ejecuta la función **comprobar()** que, evalúa si el texto es "irene", si lo es, aparecerá en objeto *ok* y si no el objeto *mal.*

```
btn_comprobar.addEventListener(MouseEvent.CLICK, comprobar);

function comprobar (Event:MouseEvent):void{

      if(nombre.text=="irene"){
            //Si lo es, se verá el movieClip verde
            ok.visible=true;
            btn_volintentar.removeEventListener(MouseEvent.CLICK,
volver);}
            //Sino la x roja
      else{
            mal.visible=true
      }
```

El segundo botón, elimina el objeto *mal*, y borrar el área de texto, en el caso en el que no se haya hecho correctamente.

```
btn_volintentar.addEventListener(MouseEvent.CLICK, volver);

function volver (Event:MouseEvent):void{
      mal.visible=false;
      nombre.text=""
}
```

5.2.14. Reproducir sonidos

Objetivo:

Al presionar una de las imágenes se oirá un sonido. El sonido será diferente, dependiendo de la imagen que se presione.

Para ello se necesitan tres imágenes que se convertirán en movieClips, que se llamarán, *mcgato, mcgallo* y *mccaballo.* Se importarán tres sonidos y se les dará un nombre de clase, para luego crear las instancias de los sonidos, que se llamarán *gatosonido, gallosonido* y *caballosonido*. Se asociará cada sonido a cada imagen, de tal forma que al hacer click en la imagen, se escuche un sonido.

Figura 5.2.18. Estado Inicial. Reproducir sonidos

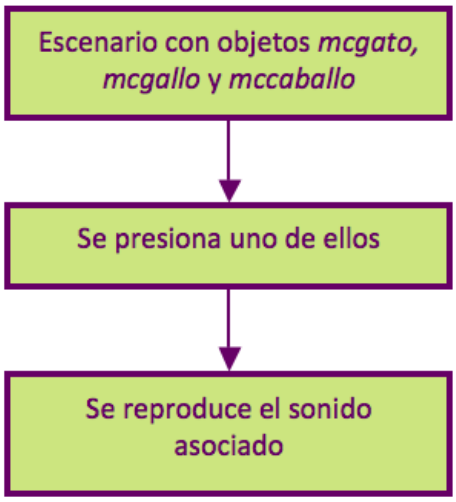


Diagrama 5.16. Reproducir sonidos

Código:

Primero se tendrán que importar los sonidos, luego en propiedades se exportarán para ActionScript 3.0 y se les dará un nombre de clase, con esto se podrá crear instancias de los sonidos.

Para crear la instancia de los sonidos: en la parte derecha se tendrá que poner el nombre de la clase que se ha creado para cada sonido y en la parte izquierda el nombre de instancia que se quiera.

```
var gatosonido:Sound = new sonido1();
var gallosonido:Sound = new sonido2();
var caballosonido:Sound = new sonido3();
```

Se crean los detectores, asociados a las funciones para que al pulsar cada dibujo, se pueda oír el sonido, gracias al método **play();**

```
mcgato.addEventListener(MouseEvent.CLICK, gato);

function gato(event:MouseEvent):void{
            gatosonido.play();
}

mcgallo.addEventListener(MouseEvent.CLICK, gallo);

function gallo(event:MouseEvent):void{
            gallosonido.play();

}

mccaballo.addEventListener(MouseEvent.CLICK, caballo);

function caballo(event:MouseEvent):void{
            caballosonido.play();

}
```

5.2.15. Dibujar.

Objetivo:

Al presionar y mover el ratón, se dibujará sobre el escenario una línea que dejará de hacerse cuando se suelte el ratón

Para ello se creará una forma, donde se dibujará, llamada *pizarra*. Con el objeto graphics, se creará la línea que se dibujará. Se posicionará en el puntero del ratón cuando se haga click, con el método **moveTo()** y se indicará que vaya haciendo la línea por donde pase el ratón, con el método **lineTo**. Para borrar, se hará doble click, que hará que se ejecute el método **clear()**; que borra lo que ha dibujado el objeto graphics. Al soltar el ratón se llama al método **removeEventListener();** asociado a la función **pintar**, por lo que se dejará de trazar la línea.

Figura 5.2.19. Estado inicial y final. . Dibujar

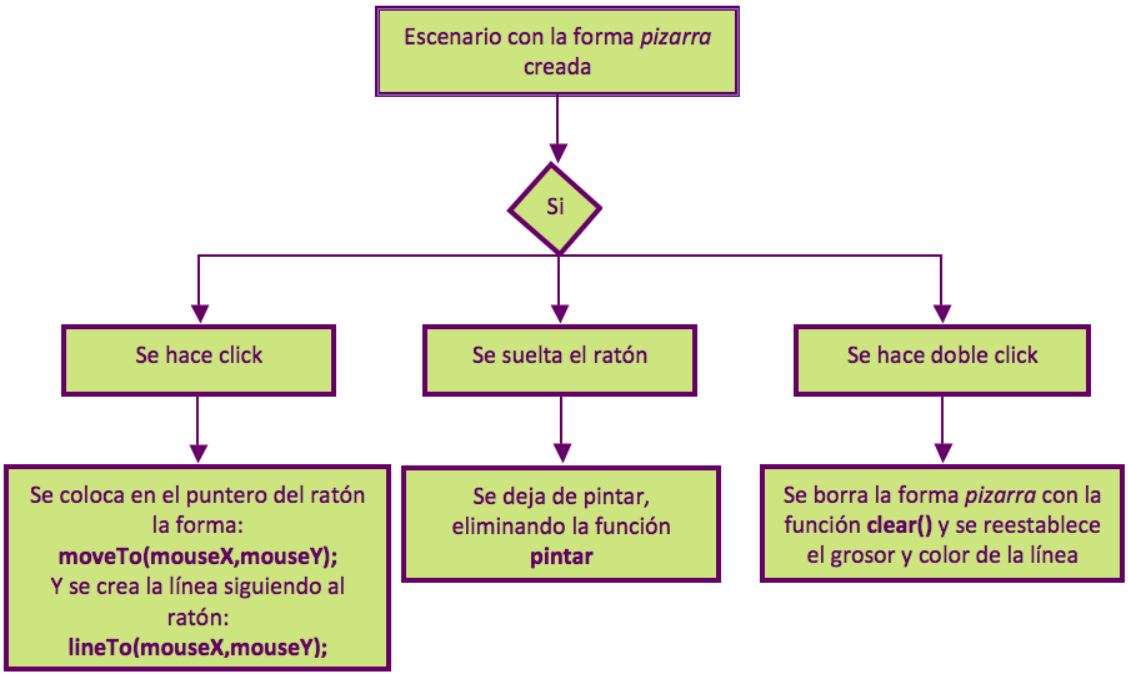


Diagrama 5.17. Dibujar

Código:

Pizarra lo que hará es contener a los dibujos.

```
var pizarra:Shape;
pizarra = new Shape();
```

Se define como va a ser el tipo de líneas en la *pizarra*. En este caso sería grosor 1, y color negro.

```
pizarra.graphics.lineStyle(1,0x000000);
addChild(pizarra);
```

Se activa el método de doble click.

```
stage.doubleClickEnabled = true;
```

Cuando se hace click, se llama a la función **sipintar()**, lo que hace es llevar el punto de inicio de ese nuevo trazo, al puntero del ratón, con el método **moveTo()**. Se añade un detector de eventos **ENTERFRAME,** asociado a la función **pintar(),** para que se pinten líneas siguiendo al puntero, con el método **lineTo(),** que lo que hace es pintar una línea hasta el puntero del ratón.

```
stage.addEventListener(MouseEvent.MOUSE_DOWN,sipintar);

function sipintar(e:Event):void{
  //Se coloca el trazo en el puntero del ratón
      pizarra.graphics.moveTo(mouseX,mouseY);
  //Se añade el evento ENTERFRAME
      addEventListener(Event.ENTER_FRAME,pintar);
}

function pintar(e:Event):void{
 //Irá trazando una línea por donde pase el ratón
      pizarra.graphics.lineTo(mouseX,mouseY);
}
```

Al soltar el ratón, se elimina el evento para pintar líneas.

```
stage.addEventListener(MouseEvent.MOUSE_UP,nopintar);
stage.addEventListener(MouseEvent.MOUSE_OUT,nopintar);

function noPintar(e:Event):void{
      removeEventListener(Event.ENTER_FRAME,pintar);
}
```

La función **clear()**; borra lo que se haya dibujado en el objeto graphics, y restablece la configuración de grosor y color de línea.

```
stage.addEventListener(MouseEvent.DOUBLE_CLICK,borrar);

function borrar(e:Event):void{
      pizarra.graphics.clear();
      pizarra.graphics.lineStyle(10,0x000000);
}
```

5.2.16. Diálogos

Objetivo:

Al presionar uno de los rectángulos con una pregunta en su interior, aparecerá en el cuadro de texto la respuesta correspondiente.

Para ello se colocarán en el escenario tres movieClips, que serán la pregunta, llamados, *hola, que* y *adios* y que harán de botones, y se creará una área de texto en la que, dependiendo de que pregunta se elija, aparecerá una respuesta.

Figura 5.2.20. Estado inicial y final.

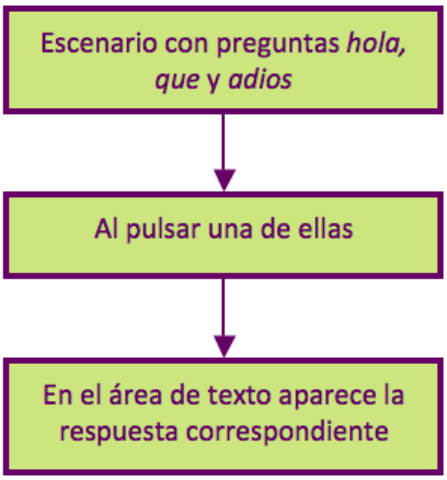


Diagrama 5.18. Diálogos

Código:

Se crea un campo de texto donde se colocarán las respuestas.

```
var respuesta:TextArea = new TextArea();

 respuesta.move(350,250);
 respuesta. (100,25);

 addChild(respuesta);
```

Se crea un escuchador para cada pregunta a la que se asocia una respuesta.

```
hola.addEventListener(MouseEvent.CLICK, respuesta1);
que.addEventListener(MouseEvent.CLICK, respuesta2);
adios.addEventListener(MouseEvent.CLICK, respuesta3);
```

Dependiendo que pregunta se elija, cada función **respuesta1()**, **respuesta2()** y **respuesta3()**, escribirá en el área de texto una respuesta diferente.

```
function respuesta1 (Event:MouseEvent):void{
      respuesta.text= "irene" ;
}

function respuesta2(Event:MouseEvent):void{
      respuesta.text= "esconderme" ;
}

function respuesta3 (Event:MouseEvent):void{
      respuesta.text= "adios" ;
}
```

5.3 MANEJO DE LA CÁMARA

El siguiente paso una vez terminadas todas las aplicaciones de Flash fue comenzar con el estudio de la cámara con la que se iban a realizar todas las grabaciones necesarias. La cámara que se usó fue la Sony PMW EX3. Para ello, hubo una lectura previa del manual de la cámara, para poder conocer todas las posibilidades que ofrecía y así poder determinar y describir las grabaciones que se realizarían.

Durante varios días se fueron realizando una serie de grabaciones para aprender a manejar la cámara y controlar la iluminación del plató para saber cual es la mejor configuración para obtener los mejores resultados en las grabaciones. Los vídeos grabados se editaron y se les aplicó el efecto Ultra key. Por lo que, tendrá que estudiarse previamente como funciona esta herramienta y los parámetros que dependen de ella, para así conseguir eliminar el fondo, con los mejores resultados posibles. Por último se realizaron unas grabaciones en las que se grabó al personaje con diferentes encuadres y movimientos. Los que obtuviesen los mejores resultados, serían los que se utilizarían para las grabaciones finales.

El primer día se realizarán una serie de grabaciones con la cámara Sony PMW-EX3 para familiarizarnos con ella (antes se habrá leído el manual de la misma) sobre los diferentes puntos que interesan:

-Balance de blancos.
-Control de ganancias.
-Diafragma.
-Velocidad de obturación.
-Distancia focal.
-Formatos de grabación

Las siguientes grabaciones se realizarán con una persona voluntaria.

Las siguientes grabaciones se realizarán con una persona voluntaria.

La iluminación que se dispondrá será una iluminación típica a tres puntos. Tal y como se detalla en la *figura 5.10.*:

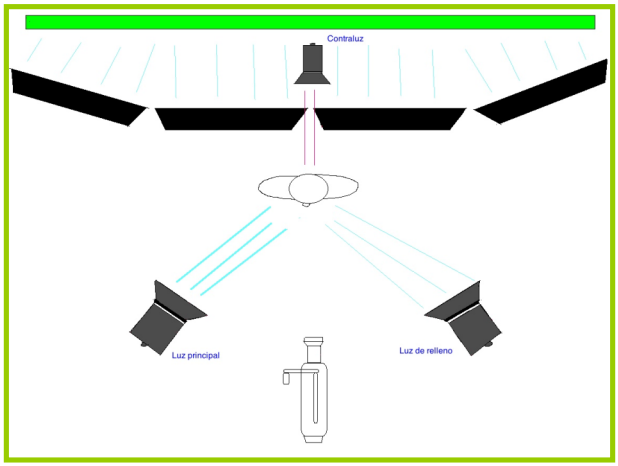


Figura 5.8. Esquema de iluminación a tres puntos para el manejo de la cámara

Todas las grabaciones se harán en HQ720/25p, ya que para lo que se quiere comprobar no hace falta usar una resolución mayor. Además así, la edición de los vídeos será más rápida que si se graba con una resolución mayor. Sin embargo, sí se realizará una grabación final en la que se probará a grabar con diferentes formatos para ver diferencias y tomar la decisión de con que formato se grabarán los vídeos finales.

Posteriormente se comprobarán los resultados editando los vídeos y exportándolos con el editor de video Adobe Premiere Pro CS5, compatible con los archivos que obtendremos de la cámara.

En la cámara realizaremos los siguientes ajustes:

-Full Auto: desactivado
-Shutter: ON
-Steady Shot: OFF
-Iris: MANUAL
-Macro: OFF
-Focus: MANUAL

ESCALETA

La siguiente tabla muestra el orden de las pruebas y el nombre de cada una de ellas.

PREPARATIVOS PLATÓ	DURACIÓN	HORA
Iluminación a 3 puntos	00:30:00	10:30
Ajustes de cámara	00:30:00	11:00
BALANCE DE BLANCOS		
Gris	00:10:00	11:10
Blanco	00:10:00	11:20
Cambio	00:05:00	11:25
GANANCIA		
6 dB	00:10:00	11:35
0 dB	00:10:00	11:45
-3 dB	00:10:00	11:55
Cambio	00:05:00	12:00
SHUTTER		
Mov. RÁPIDO 1/33	00:05:00	12:05
Mov. LENTO 1/33	00:05:00	12:10
Mov. RÁPIDO 1/50	00:05:00	12:15
Mov. LENTO 1/50	00:05:00	12:20
Mov. RÁPIDO 1/250	00:05:00	12:25
Mov. LENTO 1/250	00:05:00	12:30
Mov. RÁPIDO 1/1000	00:05:00	12:35
Mov. LENTO 1/1000	00:05:00	12:40
Cambio	00:05:00	12:45
PROFUNDIDAD DE CAMPO		
Mínima profundidad	00:20:00	13:05
Mayor profundidad	00:20:00	13:25
Cambio	00:05:00	13:30
FORMATO DE VIDEO		
HQ 1080/24p	00:10:00	13:40
HQ 720/24p	00:10:00	13:50
HQ 1080/25p	00:10:00	14:00

Al principio de la grabación el sujeto mostrará un folio que indique el nombre principal de la prueba, por ejemplo: **BALANCE DE BLANCOS** y el nombre de la subprueba, por ejemplo: **Gris.** Así, a la hora de ver los vídeos en el editor, se sabrá inmediatamente que prueba es.

Las grabaciones que se realizarán serán las siguientes:

5.3.1. Balance de blancos

Antes de empezar a grabar, se tendrá que realizar el balance de blancos, por lo que se comprobará cual es la mejor forma de realizarlo.

- **CON CARTULINA GRIS**
- **CON CARTULINA BLANCA**

Se realizará un balance. A continuación, se grabará al sujeto, se realizará el segundo balance y se grabará lo mismo.

En teoría realizarlo con una carta blanca o gris no supone ninguna diferencia, ya que los dos reflejan por igual cada componente de color, sin embargo lo realizaremos de las dos formas, para comprobar si puede existir alguna.

5.3.2. Ajustar la ganancia

Se comprobará la diferencia de grabar con diferentes niveles de ganancia. Para ello se grabará a un sujeto con un objeto oscuro (por ejemplo, una camiseta negra) que es donde se pierden más los detalles y por lo tanto donde mejor se apreciará el ruido de la imagen, así se verán las diferencias.

Se ajustarán los parámetros necesarios, para tener la misma exposición en los tres casos.

- **GANANCIA -3 dB.**
- **GANANCIA 0 dB.**
- **GANANCIA 6 dB.**

Para realizar un correcto chroma key lo mejor es no usar la ganancia de la cámara ya que eso introducirá ruido en la imagen. Esto se podrá comprobar en los resultados que se obtendrán de las grabaciones con ganancia 6dB en las cuales podrá apreciarse fácilmente el ruido de la imagen. Se compararán los resultados entre la grabación con -3 y 0 dB, ya que una ganancia negativa, lo que en realidad significa, es una atenuación de la señal de video. Esto implicará necesitar más luz para una correcta exposición pero mayor calidad de imagen.

5.3.3. Ajuste de obturador

Para el formato de vídeo que se ha elegido se tienen diferentes opciones de velocidad de obturación: 1/33; 1/50; 1/60; 1/100; 1/120; 1/125; 1/250; 1/500; 1/1000; 1/2000.
Se realizarán las siguientes grabaciones ajustando el diafragma para tener la misma exposición para todas ellas.

- **MOVIMIENTOS RÁPIDOS**
  - -Obturador a 1/33
  - -Obturador a 1/50
  - -Obturador a 1/125
  - -Obturador a 1/1000
  - -Obturador a 1/2000

- **MOVIMIENTOS LENTOS**
  - -Obturador a 1/33
  - -Obturador a 1/50
  - -Obturador a 1/125
  - -Obturador a 1/1000
  - -Obturador a 1/2000

Con las siguientes grabaciones se verán determinados efectos causados por una mal ajuste del obturador. Por ejemplo, a una velocidad baja de obturación (1/33 seg.) si se producen movimientos rápidos, en la imagen se verá una estela en todo el recorrido del movimiento. Y para una velocidad rápida de obturación (1/1000 – 1/2000 seg.) y movimientos rápidos se producirá un efecto de "strobo", es decir, de discontinuidad en la imagen, por supuesto, no se quiere que ocurran ninguno de estos dos fenómenos.

5.3.4.- Profundidad de campo

Con esta grabación se aprenderá a controlar la profundidad de campo y a ajustarla lo mejor posible a lo que se necesite en cada momento.

Para poder controlar la profundidad de campo se deberán controlar el diafragma, la distancia focal y el punto de enfoque. Para ello, será conveniente mover la cámara.

Se sabe que la profundidad de campo es muy pequeña, cuando: el diafragma está totalmente abierto, la distancia focal es muy grande y el punto de enfoque es cercano, y muy grande en condiciones contrarias.

Se realizarán estas pruebas con un sujeto moviéndose y con la ayuda de un metro se medirá la profundidad de campo para comprobar cual es la mínima y la más grande que se puede conseguir en el plató, (para referirse a ella en las pruebas, se le denominará mayor profundidad de campo), queriendo mantener al sujeto enfocado en todo momento, pero el fondo desenfocado.

- **MÍNIMA PROFUNDIDAD DE CAMPO**
  - -Diafragma totalmente abierto
  - -Distancia focal grande
  - -Cámara lo más cerca posible

- **MAYOR PROFUNDIDAD DE CAMPO**
  - -Diafragma cerrado
  - -Distancia focal pequeña
  - -Cámara lo más alejada posible

La situación de mínima profundidad de campo, se podrá usar en las pruebas de primer plano y en las que el sujeto no cambia de plano al moverse. Sin embargo, la situación de mayor profundidad de campo se calcula para cuando el sujeto se desplaza hacia la cámara, cambiando de plano americano a primer plano y viceversa. En este caso, se tendrá que buscar la profundidad de campo que abarque todo ese recorrido para que el sujeto no se salga de foco, manteniendo el fondo desenfocado.

5.3.5. Formatos de vídeo.

La cámara PMW EX3 permite una gran selección de formatos de vídeo para NTSC y PAL.

Con la opción "NTSC Area" seleccionada

HQ 1920 × 1080 59,94 Entrelazado→ HQ 1080/60i
SP 1440 × 1080 59,94 Entrelazado→ SP 1080/60i
HQ 1920 × 1080 29,97 Progresivo →HQ 1080/30P
HQ 1920 × 1080 23,98 Progresivo→ HQ 1080/24P
SP 1440 × 1080 23,98 Progresivo →SP 1080/24P
HQ 1280 × 720 59,94 Progresivo→ HQ 720/60P
HQ 1280 × 720 29,97 Progresivo→ HQ 720/30P
HQ 1280 × 720 23,98 Progresivo →HQ 720/24P

Con la opción "PAL Area" seleccionada

HQ 1920 × 1080 50 Entrelazado → HQ 1080/50i
SP 1440 × 1080 50 Entrelazado → SP 1080/50i
HQ 1920 × 1080 25 Progresivo → HQ 1080/25P
HQ 1280 × 720 50 Progresivo → HQ 720/50P
HQ 1280 × 720 25 Progresivo → HQ 720/25P

La forma de codificar estos formatos es:

En modo HQ:
-Con una compresión MPEG-2 Long GOP
-Muestreo 4:2:0
-Velocidad binaria 35Mb/s VBR

En modo SP:
-Con una compresión MPEG-2 Long GOP
-Muestreo 4:2:0
-Velocidad binaria 25Mb/s CBR

Finalmente se decidió hacer todas las grabaciones en **modo HQ**, ya que dará la mejor calidad de imagen, con las dos resoluciones que permite, para decidir si es conveniente o no grabar los vídeos finales con la máxima resolución (1920x1080) ya que, la edición y la fase de exportación será más lenta y por lo tanto podrá entorpecer el trabajo y en **progresivo** porque con este formato, se obtienen mejores resultados a la hora de eliminar el chroma. Además, se comprobarán las diferencias de grabar con un número diferente de fotogramas.

- **GRABAR CON HQ1080/24p**
- **GRABAR CON HQ 720/24p**
- **GRABAR CON HQ 1080/25p**

Estas pruebas se realizarán en el plató de televisión. Se calcula que a lo largo de una mañana, es decir de 10:00 a 14:00 horas.

5.3.6. Análisis de los resultados

Los vídeos de los resultados obtenidos, se encuentran en el DVD en la carpeta **Pruebas > 5.3. MANEJO DE CAMARA**

- **Balance de blancos**

El balance de blancos se realizó de dos formas. Para el balance con color gris, se usó un folio blanco y se subexpuso la cámara hasta que el nivel de vídeo fue del 50% (esto se comprobó con el patrón zebra de la cámara) que es el nivel del gris medio y para el balance con color blanco se usó un folio blanco también, en este caso la cámara se subexpuso a un nivel de vídeo de 80%, esto es recomendable hacer para que no se produzca saturación en los colores primarios y el balance de blancos sea incorrecto.

En los dos casos se obtuvo un valor de 3400K, se puede ver en el vídeo **balance.flv,** aunque cada uno tiene un valor de exposición diferente. Este valor se mantuvo para el resto de las pruebas.

- **Ganancia**

Para comprobar las diferencias de grabar con distintos valores de ganancia se mantuvo la velocidad de obturación en 1/50 y con los diferentes valores de ganancia, -3dB, 0dB y 6dB se fue modificando el diafragma para tener la misma exposición en cada vídeo.

En el video **ganancia.flv** se puede apreciar las diferencias entre usar una y otra, sobre todo en la camiseta negra, donde se ve más ruido cuando se usa la ganancia de 6dB, sin embargo, se ve que con 0dB y -3dB se ha minimizado notablemente.

- **Shutter**

En estas pruebas se fue modificando el diafragma para tener la misma exposición en cada vídeo.

En el vídeo **movlentos.flv**, se ve el efecto de grabar movimientos lentos con diferentes valores de shutter, se comprueba que no existe ningún problema ni efecto molesto, para ningún valor. En cambio en el vídeo **movrapidos.flv** se aprecian los efectos de grabar movimientos rápidos. Con velocidades bajas, se ve una estela en todo el movimiento, y con velocidades altas, la mano se ve perfectamente en todo momento, esto da sensación de discontinuidad (efecto strobo). Sin embargo, el efecto de estela, es más natural para el ojo humano, no es molesto. Las diferencias se aprecian muy bien en los videos **shutter_50_1000.flv** y **shutter_33_2000.flv.**

- **Profundidad de campo.**

Para la profundidad mínima se grabó a F1,9, 1/50 y -3dB y distancia focal 77 mm. La distancia enfocada por delante del sujeto que se obtuvo, fue de unos 50cm. Estos serán los valores que se usarán en las grabaciones en las que el sujeto no cambia la posición

Para la mayor profundidad se grabó a F5 1/100 y 0dB y distancia focal 77mm. La distancia enfocada por delante del sujeto que se obtuvo en este caso, fue de un metro aproximadamente. Se grabó con estos niveles, ya que era con los que se obtenía menor distancia enfocada por detrás del sujeto y con la que se podía obtener una buena exposición al

grabar con 0dB de ganancia, estos valores servirán para las grabaciones donde el sujeto cambia de posición.

Las diferencias se pueden ver en el vídeo **profundidad2.flv**

- **Formatos**

Se comprobó que existen diferencias al grabar en HQ720/25p y HQ1080/25p. El formato HQ720/25p es más sensible a la luz, debido a que el sensor de la cámara es de 1920x1080 píxeles efectivos, esto significa, que al grabar con la misma resolución que el sensor de la cámara, es decir a 1080/25p, la luz que se recoge en el sensor es la misma que se ve en la imagen, sin embargo al grabar con una resolución menor, es decir a 1024x720 los píxeles son de mayor tamaño por lo que recogen más luz, por eso aparece una exposición mayor en los vídeos con esta resolución. Este efecto se puede ver en el vídeo **formatos.flv**

Finalmente en el vídeo **formatos.flv** se ven las consecuencias de grabar con los formatos HQ 720/24p y HQ1080/24p. Los cambios de luz en la imagen son debidos a que los fluorescentes parpadean a una frecuencia de 50Hz (esta frecuencia viene dada por los valores de la red de alimentación eléctrica que existe en España que es de 230V y 50Hz) y la frecuencia de los vídeos es de 24 fotogramas por segundo.

La explicación de este efecto se puede ver gráficamente en el siguiente dibujo:

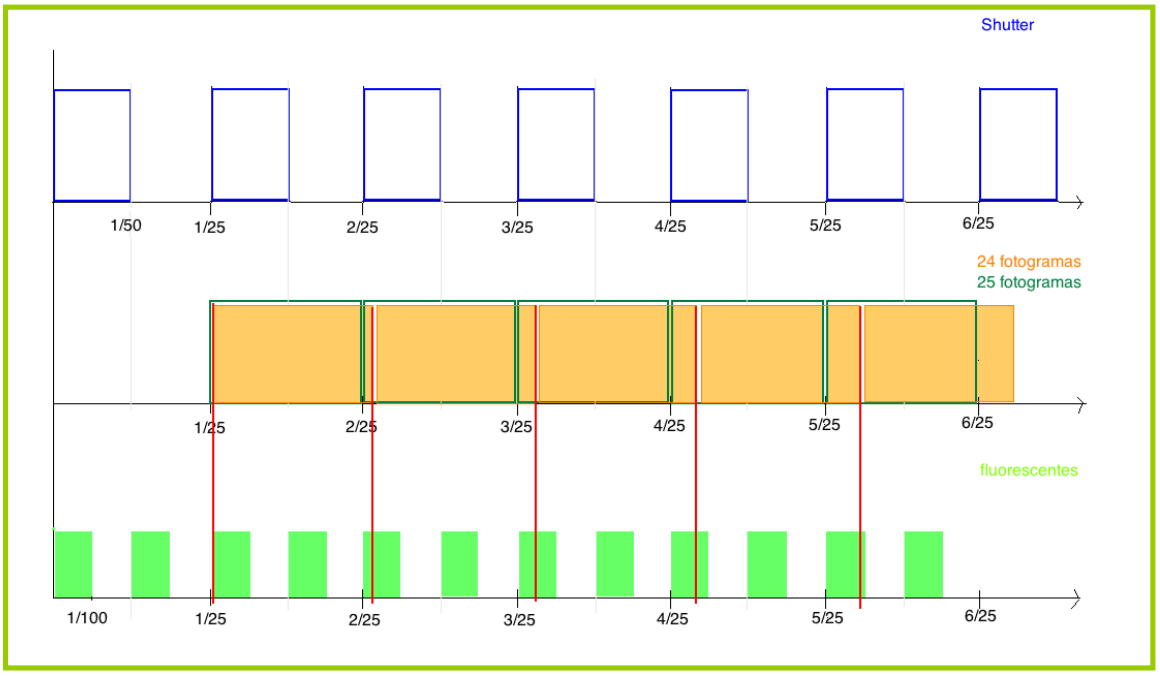


Figura 5.9. Esquema de los periodos del shutter, fotogramas y fluorescentes

Los periodos de parpadeo del fluorescente no coinciden con los tiempos de los fotogramas cuando se usan 24 fotogramas. En cada fotograma tenemos diferente cantidad de luz, por eso se ven cambios. Sin embargo cuando usamos un formato de 25 fotogramas, si que coinciden los periodos de parpadeos de los fluorescentes, por eso no hay cambios en la iluminación.

5.4 ILUMINACIÓN Y GRABACIÓN

Después de hacer las pruebas del manejo de cámara se diseñaron las del iluminación. Con estas se pretende definir el esquema de iluminación que va a utilizarse en las grabaciones finales. Para ello, se propuso utilizar el típico esquema de iluminación a tres puntos y hacer modificaciones según se creyera necesario. En las pruebas anteriores ya se trabajó con este esquema de iluminación y dio buenos resultados. Sin embargo, a estos vídeos se les aplicará el efecto *Ultra Key* para ver si es posible eliminar el fondo correctamente con esa iluminación y con las variaciones que se realicen.

Con estas grabaciones se aprenderá a dominar la iluminación del plató, y ver que resultados se obtienen al grabar con variaciones en la iluminación. Esto, junto con los resultados de la grabaciones anteriores, permitirá conseguir los mejores resultados en las grabaciones finales.

Los vídeos que se graben, serán posteriormente editados para eliminar el chroma y así poder analizar los resultados de una buena o mala iluminación. Por lo tanto, se explicará este proceso, profundizando en cada parámetro. También, se explicará la forma de exportar el vídeo para poder ser visualizado en Flash con el fondo transparente y obtener la mejor calidad con la codificación.

El segundo día se realizarán las grabaciones con una iluminación a tres puntos y se harán los cambios necesarios a la hora de grabar las diferentes pruebas, como:

-Ajustar el contraluz cuando pasemos de la prueba de primer plano a plano americano.

-Colocación de un segundo contraluz para las pruebas finales.

Para estas pruebas se decidió usar principalmente el formato HQ720/25p ya que, como se ha visto en las pruebas anteriores, los resultados en este formato son muy buenos, aunque también se hizo una prueba en HQ1080/25p para ver las diferencias de eliminar el chroma, con uno y otro.

Cada prueba se grabará de tres maneras diferentes:

-F1,9 1/500 -3dB
-F1,9 1/250 -3dB
-F2,8 1/250 -3dB

Estos valores son aproximados, se modificarán según se necesite, pero se consideró, después de realizar las pruebas de cámara, que son las mejores combinaciones para obtener buenos resultados.

Se usó un diafragma muy abierto para tener poca profundidad de campo y -3dB de ganancia para obtener mejor calidad de imagen.

Además cada una de las pruebas se hará con el contraluz magenta y sin él, para ver si existe alguna diferencia significativa a la hora de eliminar el chroma.

Los planos y movimientos elegidos para las pruebas, son debido a que serán los encuadres y movimientos básicos que se usarán para las grabaciones finales. Las grabaciones que se realizaron son las siguientes:

5.4.1.- Primer plano.

- **De frente**
- **De perfil**

La iluminación en estos casos se efectuará con un fluorescente haciendo de luz principal, colocado a la izquierda del sujeto a unos 45º de la cámara y el otro fluorescente haciendo de luz de relleno colocado en la derecha a unos 45º también pero con menor intensidad de luz. El contraluz se realizará con un fresnel situado detrás del actor y arriba, a unos 45º, colocaremos un filtro color magenta en el contraluz, para contrarrestar las posibles sombras verdes provenientes del fondo, que puedan aparecer en el sujeto.

Para estás pruebas se dispondrá el siguiente esquema de iluminación:

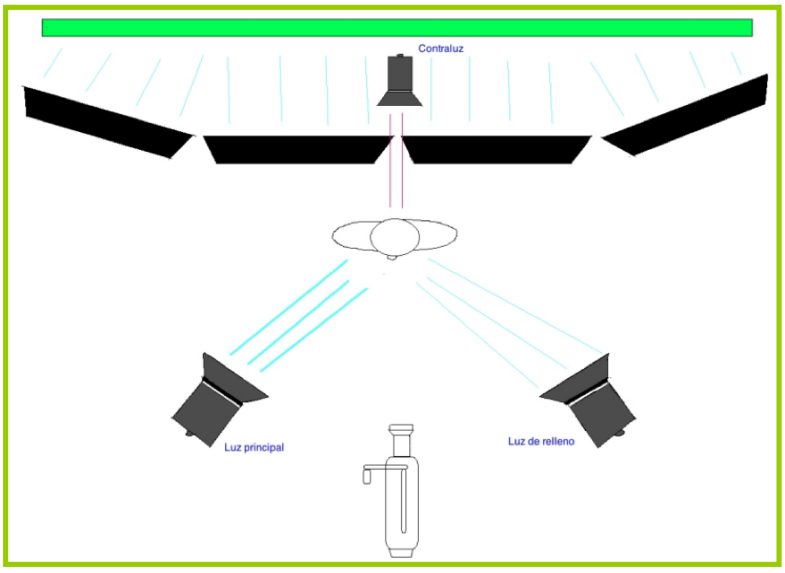


Figura 5.10. Esquema de iluminación para las pruebas de primer plano

5.4.2.- Misma posición.

- De frente
- De perfil

Se dispondrá el mismo esquema de iluminación salvo que, al cambiar el plano en estas pruebas tendremos que modificar el contraluz para evitar brillos en zonas no deseadas.

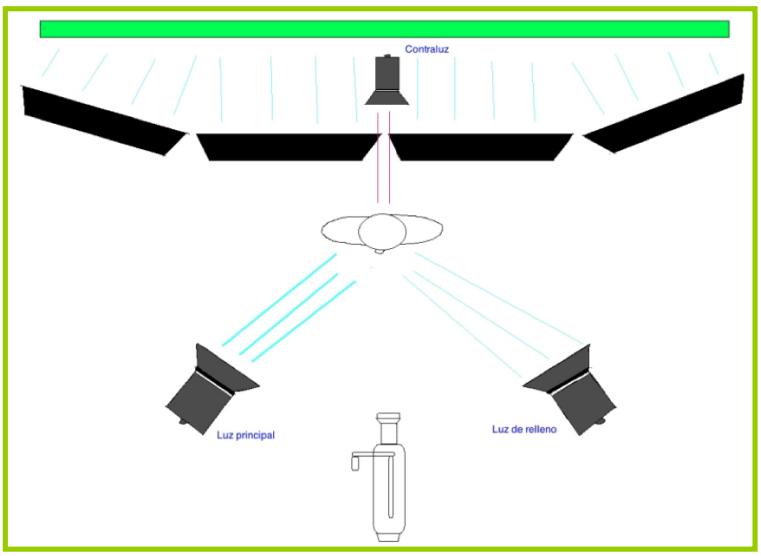


Figura 5.11. Esquema de iluminación para las pruebas del sujeto en la misma posición

Estas pruebas se grabarán también en formato HQ1080/25p para ver si existen diferencias a la hora de eliminar el chroma con un formato de mayor calidad. Se eligió grabar estas pruebas en este formato y no en otro, porque en ese momento, se pensó que para la aventura gráfica, lo más conveniente y los mejores resultados, se obtendrían de las grabaciones del personaje en la misma posición y que de esta forma se grabarían los vídeos finales. En principio no hay duda de que será más fácil eliminar el chroma con este formato, pero interesa comprobar si las diferencias son significativas.

5.4.3.- Diferente posición

De frente

Se usará el mismo esquema que en las pruebas anteriores.

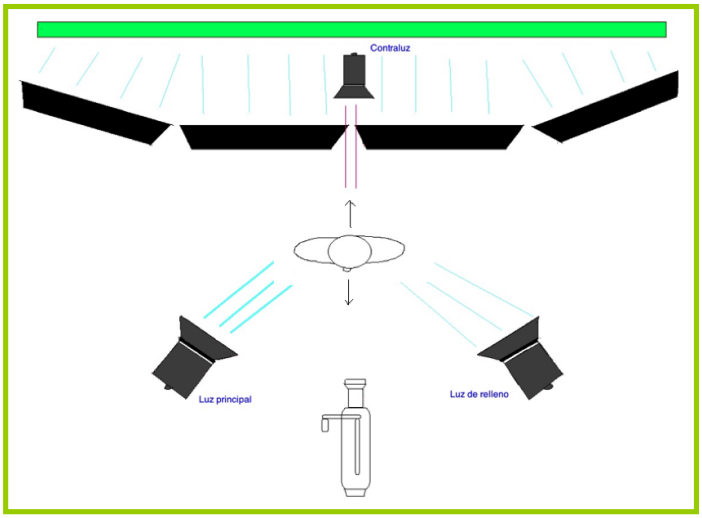


Figura 5.12. Esquema de iluminación para las pruebas del sujeto en diferente posición

De perfil

Para las pruebas en las que el personaje se mueve hacia los lados, se mantendrá el mismo esquema pero se colocará otro contraluz, para cubrir toda la zona. En principio no se cambiarán de posición los demás focos, debido a que el desplazamiento del sujeto será el mínimo posible.

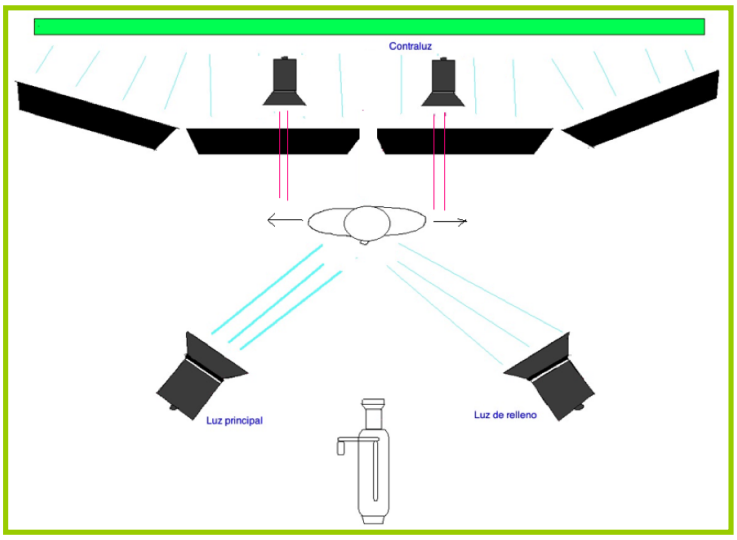


Figura 5.13. Esquema de iluminación para las pruebas del sujeto en diferente posición

Cada toma será de 10 segundos y se grabará dos veces por lo menos.

Estas pruebas se realizarán a lo largo de una mañana, es decir de 10:00 a 14:00 horas.

5.4.4 Edición de vídeos

Antes de proceder con el análisis de resultados, se explicará el proceso de la edición y exportación de los vídeos.

Los vídeos elegidos para trabajar son los que tienen los parámetros alrededor de F2,8 1/250 -3dB ya que son los que mejor valor de exposición tenían y a los que se les aplicó el efecto para suprimir el fondo.

El efecto que se usó en Adobe Premiere CS5 para eliminar el chroma es el *Ultra key*, situado en la carpeta Keying de efectos de vídeo. Este efecto debe arrastrarse hasta el clip de vídeo que se desea editar.

Con la herramienta cuentagotas, *figura 5.14.* se elige la clave de color, es decir, el color que se quiere eliminar. A esta zona de la imagen se le denominará canal alfa.

Figura 5.14. Captura del editor de videos de Adobe Premiere CS5

A continuación se modifican los diferentes parámetros hasta conseguir el efecto deseado. Estos parámetros son **[C7-C8]** :

Matte generation

Transparency: Controla la transparencia de la imagen cuando se ha seleccionado la clave de color. El rango de valores oscila entre 0 y 100. 100 es totalmente transparente. 0 es opaco. El valor predeterminado es 45.

Highlight: Aumenta la opacidad de las zonas claras de la imagen de origen. El rango de valores oscila entre 0 y 100. El valor predeterminado es 50. 0 no afecta a la imagen.

Shadow: Aumenta la opacidad de las zonas oscuras de la imagen de origen. El rango de valores oscila entre 0 y 100. El valor predeterminado es 50. 0 no afecta a la imagen.

Tolerance: Aumenta la tolerancia a la variación a partir del color elegido como clave de color. La función. El rango de valores oscila entre 0 y 100. El valor predeterminado es 50. 0 no afecta a la imagen.

Pedestal: Filtra el ruido en el canal alfa. El rango de valores oscila entre 0 y 100. El valor predeterminado es 10. 0 no afecta a la imagen. Cuanto más elevada sea la calidad de la imagen de origen, más bajo podrá establecer el Pedestal.

Estos parámetros conviene usarlos con la imagen en tipo *Alpha Channel*, en la opción de *Output, figura 5.15*, este tipo de imagen, muestra las partes transparentes en negro y las opacas en blanco. Por lo tanto modificando estos parámetros se deberá conseguir que la zona de la persona sea completamente blanca y el fondo completamente negro.

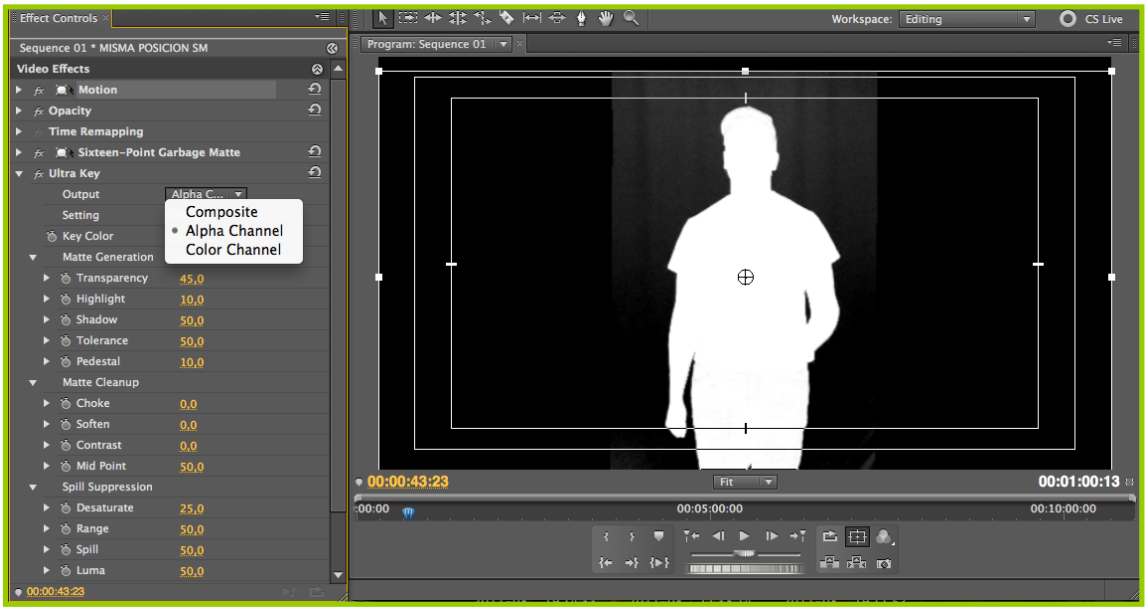


Figura 5.15. Imagen en modo Alpha Channel

En esta imagen se aprecia que el fondo no es del todo transparente, sino que tiene zonas en la que se verían errores. Al modificar los parámetros debe quedar algo como lo mostrado en la *figura 5.16*:

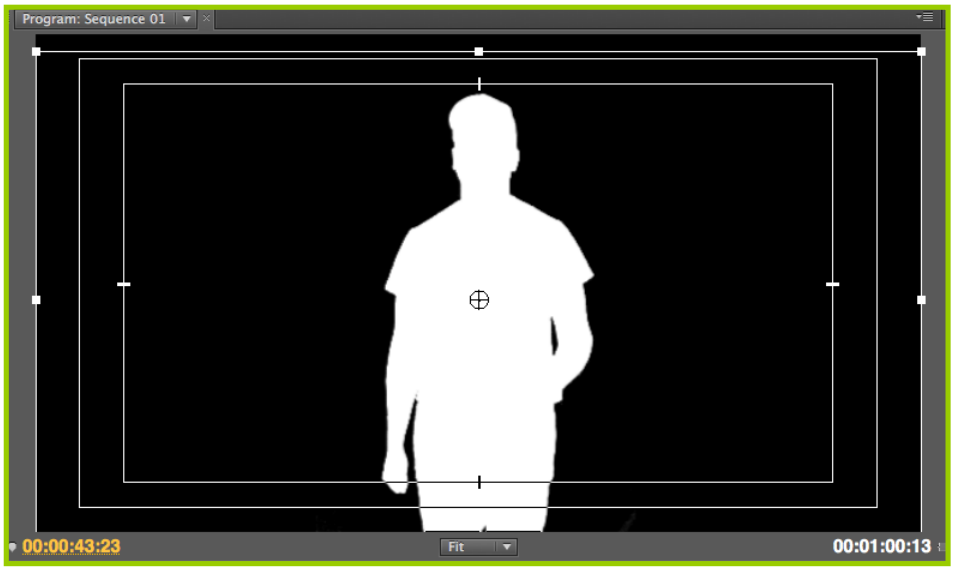


Figura 5.16. Imagen con el fondo totalmente transparente

Una vez se ha conseguido eliminar el fondo, se comprobará el resultado en la imagen, seleccionando el tipo de imagen *Composite.*

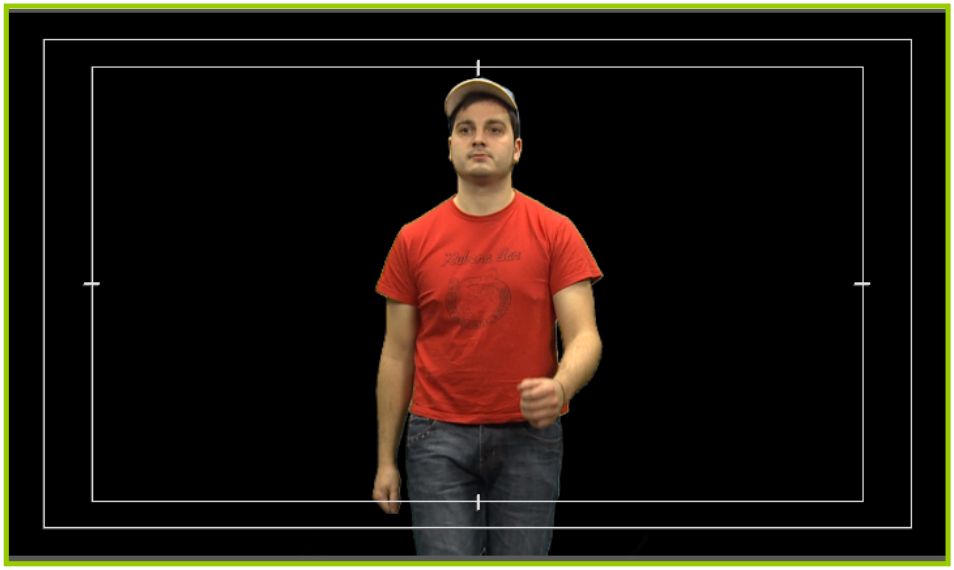


Figura 5.17. Imagen en modo Composite

Se aprecia que alrededor del personaje hay un contorno verde, *figura 5.17.*, para solucionar esto, se deben modificar los parámetros siguientes:

Mate cleanup

Choke: Reduce el tamaño del mate del canal alfa. Realiza una erosión morfológica. Los valores de nivel de retracción se encuentran entre 0 y 100. El valor predeterminado es 0.

Soften: Desenfoca el borde del mate del canal alfa. Realiza un filtro de desenfoque de cuadro. Los valores de nivel de desenfoque se encuentran entre 0 y 100. 0 no afecta a la imagen. El valor predeterminado es 0.

Contrast: Ajusta el contraste del canal alfa. El rango de valores oscila entre 0 y 100. 0 no afecta a la imagen. El valor predeterminado es 0.

Mid point: Selecciona el punto de equilibrio para el valor de contraste. El rango de valores oscila entre 0 y 100. 0 no afecta a la imagen. El valor predeterminado es 50.

En la *figura 5.18.* se puede apreciar el resultado obtenido:

En la *figura 5.18*. se aprecia una notable mejora en el vídeo, el contorno ahora es casi imperceptible. Con estas modificaciones se obtienen unos resultados muy buenos. El resto de parámetros que a continuación se describen, no será necesario modificar.

Spill supression

Desaturate: Controla la saturación del color de fondo del canal de color. Elimina la saturación de los colores que están cercanos a ser completamente transparentes. El rango de valores oscila entre 0 y 100. 0 no afecta a la imagen. El valor predeterminado es 25.

Range: Controla la cantidad de rebase que se corrige. El rango de valores oscila entre 0 y 100. 0 no afecta a la imagen. El valor predeterminado es 50.

Spill: Ajusta la cantidad de compensación de rebase. El rango de valores oscila entre 0 y 100. 0 no afecta a la imagen. El valor predeterminado es 50.

Luma: Afecta al canal alfa para restaurar la luminancia original de la imagen de origen. El rango de valores oscila entre 0 y 100. 0 no afecta a la imagen. El valor predeterminado es 50.

Los últimos parámetros afectan a la saturación, tono y luminancia de la imagen, que no interesa modificar, son los siguientes:

Color correction

Saturation: Controla la saturación de la imagen. El rango de valores oscila entre 0 y 200. 0 elimina toda la cromacidad. El valor predeterminado es 100.

Hue: Controla el tono de la imagen. El rango de valores oscila de -180° hasta +180 °. El valor predeterminado es 0°.

Luminance: Controla la luminancia de la imagen. El rango de valores oscila entre 0 y 200. 0 es negro. El valor predeterminado es 100.

Los parámetros más importantes son los que están dentro de los grupos **Matte generation** y **Matte cleanup**. Los primeros afectan a la imagen para poder eliminar mejor el fondo y los segundos, afectan a los bordes de la imagen una vez tenemos el fondo totalmente transparente.

5.4.5 Exportación de vídeo

Estos vídeos se editaron con el editor Adobe Premiere Pro CS5, ya que acepta los archivos de la cámara. Una vez editados, se exportaron en formato FLV, ya que es el formato que usa Flash para reproducir vídeo. En las opciones de exportación se elegirán los parámetros que ofrezcan la mejor calidad.

Para exportar se elige:

File > Export > Media y en *Export settings* se elegirán las opciones que vemos en la imagen siguiente:

Se elegirán las opciones de exportación que proporcionen la mejor calidad.

Lo primero que se debe elegir es el formato en el que se exportarán los vídeos, como ya se ha dicho, será en **FLV.**

Flash también admite el formato F4V, pero la diferencia, es que el códec de este formato, MainConcept H.264 Video, no permite la codificación de un canal alfa, que es necesario para exportar vídeos, con el fondo suprimido es decir, transparente.

El códec que utiliza el formato FLV es el **On2 VP6** y que admite el uso de un canal alfa de 8 bits para componer vídeo.

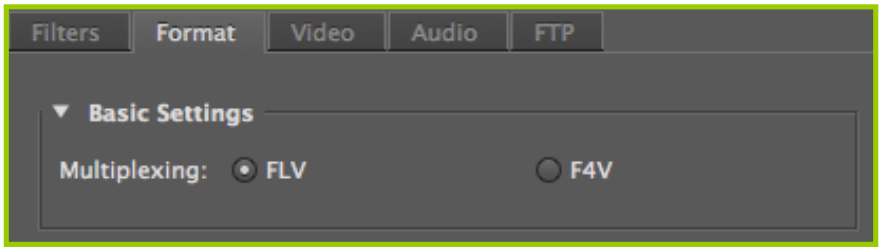


Figura 5.19. Ventana de elección del formato de vídeo

A continuación se elegirán los parámetros que ofrezcan la mejor calidad. Los elegidos se muestran en las imágenes que acompañan a la explicación de los parámetros **[V8]:**

Basic Video Settings

Encode Alpha Channel: permite elegir si se codifica el canal alfa o no.

Frame Height: Especifica la anchura del fotograma de la salida en píxeles.

Frame Width: Especifica la altura del fotograma de la salida en píxeles.

Frame rate [fps]: Velocidad de fotogramas del archivo de salida indicada en fotogramas por segundo. Se elegirá el mismo que el archivo fuente, es decir 25 fotogramas.

Render at Maximum Depth: Especifica si Adobe Premiere procesa secuencias con una profundidad de bits alta y completa.

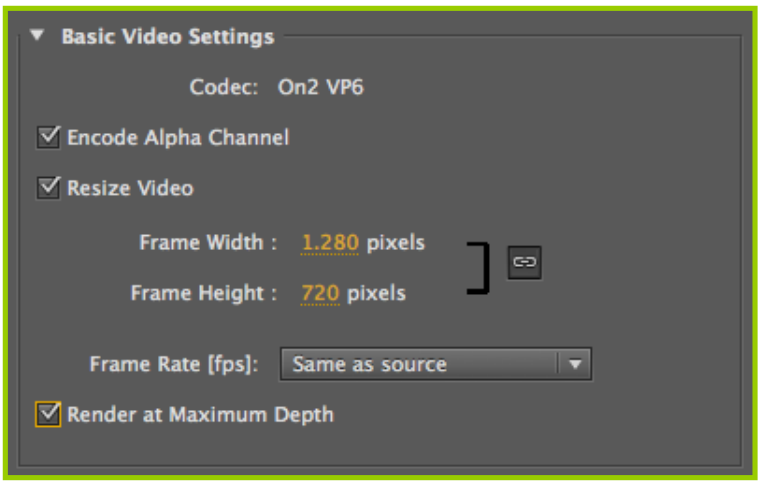


Figura 5.20. Ventana con los ajustes básicos de vídeo

Bitrate Settings

Bitrate Encoding: Especifica si el códec produce una velocidad de bits constante (CBR) o variable (VBR) en el archivo exportado:

CBR: Comprime cada fotograma del vídeo de origen con el límite fijo especificado y genera un archivo con velocidad de datos fija. De este modo, los fotogramas que contienen datos más complejos se comprimen más y los fotogramas menos complejos se comprimen menos.

VBR: Permite modificar la velocidad de datos del archivo exportado. Debido a que una cantidad determinada de compresión degrada la calidad de una imagen compleja más que la de una imagen sencilla, la codificación VBR comprime los fotogramas complejos menos y los fotogramas sencillos más.

Encoding Passes: Especifica el número de veces que el codificador analizará el clip antes de codificarlo. Si se realizan varios pases aumenta el tiempo que se tarda en codificar el archivo pero generalmente produce una compresión más eficaz y una imagen con mayor calidad.

One: El codificador aplica un solo pase al archivo, desde principio a fin. La codificación de un solo pase tarda menos tiempo que la codificación de dos pases, pero no logra la misma calidad en la salida.

Two: el codificador aplica dos pases al archivo, desde principio a fin, y seguidamente desde fin a principio. El segundo pase alarga el proceso, pero garantiza una mayor eficiencia de la codificación y, a menudo, una mayor calidad de salida.

Bitrate Level: Cuando el nivel de la velocidad de bits está definido como Personalizado, la velocidad de bits de salida se puede cambiar a cualquier valor. Cuando el nivel de velocidad de bits está definido como Alto, Medio o Bajo, la velocidad de bits se ajusta automáticamente según las dimensiones de fotograma como un valor de Sólo lectura y no se puede cambiar.

Las opciones siguientes aparecen sólo si se selecciona VBR como opción de la codificación de velocidad:

Minimum Bitrate [%target]: Especifica el número mínimo de megabits por segundo de reproducción que se desea que el codificador permita. La velocidad de bits mínima es diferente para cada formato.

Maximum Bitrate [%target]: Especifica la velocidad de transferencia máxima que debe permitir el codificador.

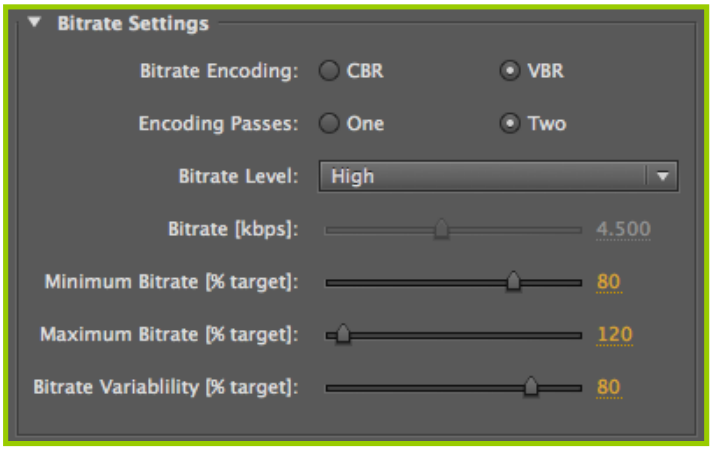


Figura 5.21. Ventana con los ajustes de velocidad de bits

Advanced Settings

Set Key Frame Distance: Número de fotogramas tras el cual el códec creará un fotograma clave cuando exporte vídeo.

Simple Profile: Al seleccionar Perfil simple se optimiza el contenido de vídeo de alta resolución que se reproducirá en equipos informáticos más antiguos o en otros dispositivos con memoria o recursos de procesamiento limitados.

Undershoot [% target]: Esta opción permite especificar el porcentaje de la velocidad de los datos de destino que se desea lograr de forma que los datos adicionales estén disponibles en el búfer para mejorar las secciones difíciles.

Quality: Esta opción permite especificar un balance entre la calidad de c odificación y el tiempo que se tarda Adobe Premiere Pro en codificar vídeo.

Good: Busca un balance entre calidad de imagen y el tiempo que se tarda en codificar vídeo. Se trata del valor predeterminado.

Best: Crea la mejor calidad de imagen posible, pero llevará bastante más tiempo codificar vídeo.

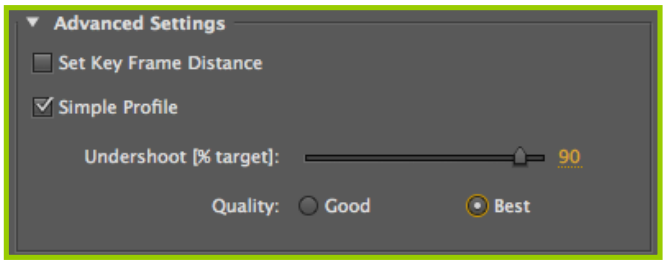


Figura 5.22. Ventana con los ajustes avanzados

5.4.6. Análisis de resultados

Los vídeos con los resultados pueden verse en la carpeta del DVD **Pruebas > 5.4 PRUEBAS DE ILUMINACIÓN Y GRABACIÓN**

Se hicieron pruebas con contraluz magenta y sin él, aunque al analizar los vídeos, no se aprecia ninguna diferencia entre ambos. Esto se puede observar en el vídeo **primerplano_magenta.flv** que muestran el primer plano del sujeto a la izquierda con contraluz magenta y a la derecha, sin él. Lo mismo ocurre con el resto de los vídeos, e las diferentes posiciones **mismaposi_magenta.flv, diferposi_magenta.flv.** A pesar de esto, el chroma se eliminó sin problemas en los dos casos con muy buenos resultados

Por último se probó a grabar con el formato HQ1080/25p para comprobar si las diferencias, a la hora de eliminar el chroma, eran significativas. Uno de los inconvenientes es que al tener mayor tamaño, se tiene que renderizar a menudo para poder ver los resultados de las modificaciones. Esto tampoco es un gran problema ya que los vídeos finales que se graben serán cortos. En los vídeos, **formato_negro.flv** y **formato_fondo.flv** se puede ver lo que se obtuvo. Se hizo una comparación con el mismo vídeo, utilizando los mismos parámetros de Ultra key, con un fondo negro y un fondo con colores respectivamente. En el vídeo que posee el fondo negro, se pueden apreciar pequeñas faltas en los bordes del personaje. Sin embargo, estas faltas desaparecen completamente al incluir un fondo con colores.

Con ese formato se obtienen muy buenos resultados sin realizar muchos cambios, y se ha comprobado que no existen problemas a la hora de trabajar con él, por lo que en las grabaciones finales del personaje, se usará este formato de vídeo.

5.5. REALIZACIÓN DEL CHROMA KEY

Terminadas las grabaciones de iluminación y junto con las de manejo de cámara, ya se tiene un control completo de la cámara y de la iluminación del plató, por lo que puede procederse a la grabación del personaje con diferentes encuadres y movimientos. Los que mejor resultados den una vez extraído el fondo, se elegirán para las grabaciones finales. Estas pruebas se grabaran a lo largo de dos días. El primer día se grabarán las pruebas de primer plano y plano americano y el segundo día las pruebas de cuerpo entero.

El **primer día** se realizaran tres tipos de pruebas que son:

Las del **primer tipo** serán con el **sujeto en la misma posición**, haciendo diferentes movimientos:

-Simular andar hacia la derecha.
-Simular andar hacia la izquierda.
-Simular andar hacia delante.
-Simular andar de espaldas a la cámara.

Las del **segundo tipo** serán con el **sujeto cambiando de posición**, haciendo diferentes movimientos:

-Andar hacia la derecha.
-Andar hacia la izquierda.
-Andar hacia delante (acercarse a la cámara).
-Andar de espaldas a la cámara (alejarse de la cámara).

Las del **tercer tipo** serán de un **primer plano del sujeto**:

-De frente.
-De espaldas.
-De los dos perfiles.

El **segundo día** se realizarán las pruebas de **cuerpo entero**:

-De frente en la misma posición.
-De perfil en la misma posición.
-De perfil cambiando de posición.
-Saltando, exagerando movimientos con los pies.

La iluminación del fondo será común para todas ellas. Sin embargo, la iluminación del sujeto es posible que haya que modificarla ligeramente, dependiendo de los movimientos que tenga que realizar este. Estos cambios, cuando los haya, vendrán detallados en la prueba.

El sujeto deberá colocarse por delante de los fluorescentes que iluminan el fondo para que no se produzcan sombras en el fondo ni reflejos en el sujeto.

El sujeto no deberá llevar ropa verde, para no interferir con el fondo. A ser posible, vestirá con colores complementarios, como el rojo y llevará una gorra, para evitar problemas en postproducción con el pelo, ya que suele ser problemático.

Los planos del sujeto del primer día serán planos americanos. En las pruebas que el sujeto tiene que acercarse y alejarse, serán de plano americano a un primer plano y viceversa.

Lo primero que se tendrá que hacer antes de empezar a grabar las pruebas será el balance de blancos, que se mantendrá para todas ellas.

Para todas las pruebas se mantendrá al sujeto enfocado, el obturador de la cámara abierto al máximo, y la distancia focal lo más grande posible.

5.5.1. Misma posición

La iluminación en estos casos se efectuará con un fluorescente haciendo de luz principal, colocado a la izquierda del sujeto a unos 45º de la cámara y el otro fluorescente haciendo de luz de relleno colocado en la derecha a unos 45º también pero con menor intensidad de luz. El contraluz se realizará con un fresnel situado detrás del actor y arriba, a unos 45º, colocaremos un filtro color magenta en el contraluz, para contrarrestar las posibles sombras verdes provenientes del fondo, que puedan aparecer en el sujeto.

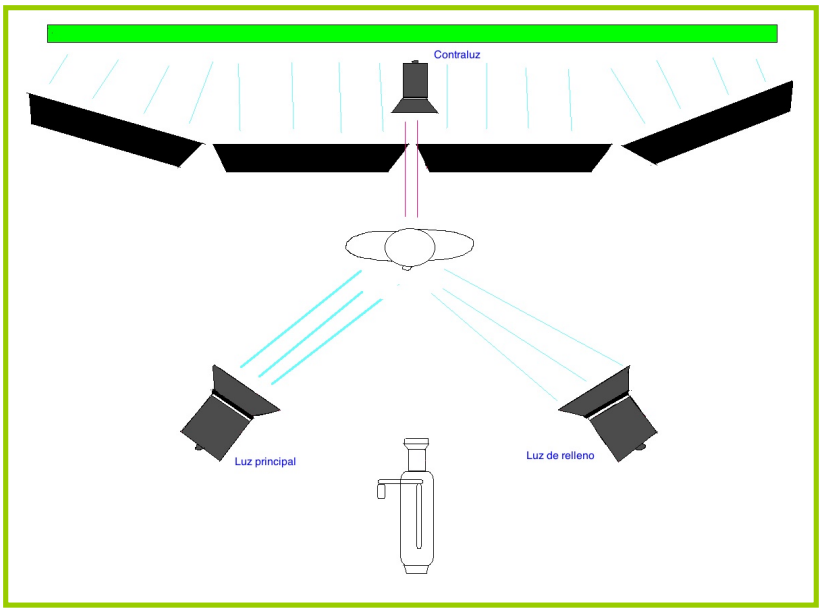


Figura 5.23. Esquema de iluminación para el sujeto en la misma posición

5.5.2. Diferentes posiciones

Para las siguientes pruebas, se mantendrá el esquema de iluminación de las anteriores. El sujeto se desplazará el mínimo espacio posible para evitar problemas de iluminación.

- **Moviéndose de izquierda a derecha y viceversa, en el mismo plano.**

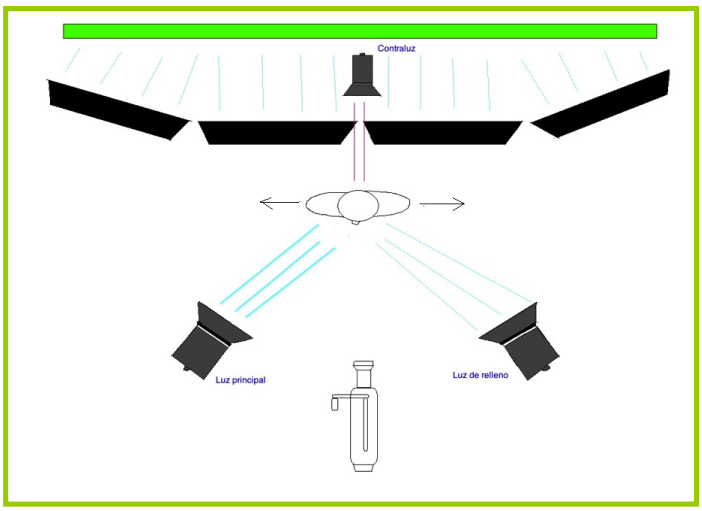


Figura 5.24. Esquema de iluminación para el sujeto moviéndose de izquierda a derecha

- **Acercándose a la cámara y alejándose**

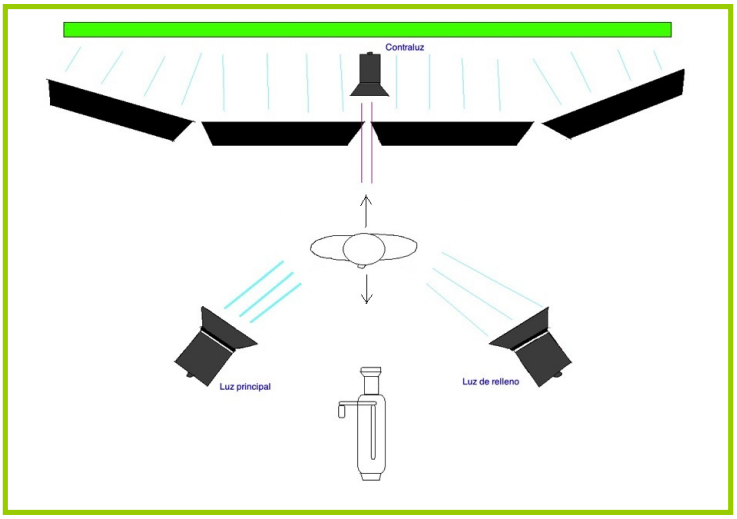


Figura 5.25. Esquema de iluminación para el sujeto acercándose a la cámara y alejándose

Si fuese necesario debido a que los resultados no fueran los esperados, se realizarían otras pruebas, modificando el esquema de iluminación.

5.5.3. Primer plano

Esta prueba se realizará con el mismo esquema de iluminación que el primer tipo de pruebas.

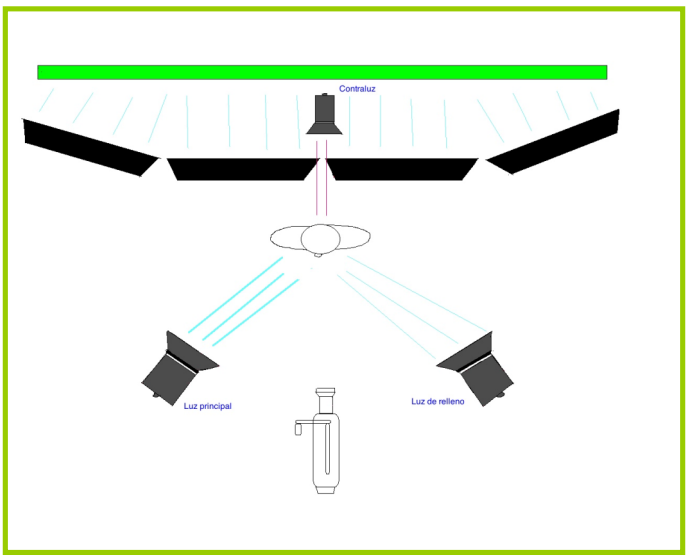


Figura 5.26. Esquema de iluminación para el primer plano del sujeto

El plano será más cerrado, de forma que solo encuadremos la cara del sujeto, manteniendo el fondo desenfocado. Habrá que modificar ligeramente la inclinación del contraluz.

5.5.4. Cuerpo entero

El segundo día se realizarán las pruebas de cuerpo entero para comprobar si es posible obtener buenos resultados eliminando el chroma de la parte de los pies.

Las pruebas se realizarán con el mismo esquema de iluminación que las pruebas del primer y segundo tipo(misma posición y diferente posición) del primer día.

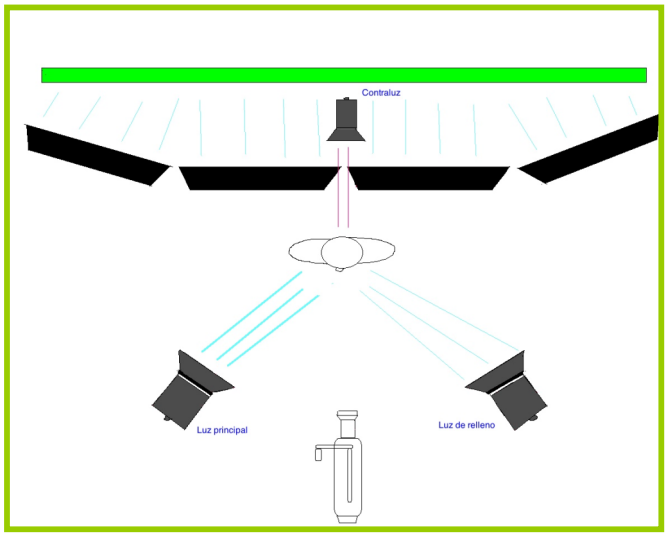


Figura 5.27. Esquema de iluminación para el sujeto grabado de cuerpo entero

Para estas pruebas se tendrá que colocar la cámara más alejada, de forma que se pueda encuadrar todo el cuerpo del sujeto.

Habrá que tener especial cuidado con la tela del chroma, ya que el sujeto deberá colocarse encima de la misma para poder grabar los pies.

Se grabará con formato HQ 1080/25p ya que con una mayor calidad se obtienen mejores resultados a la hora de eliminar el chroma, como se ha podido comprobar en las pruebas anteriores, ya que se crearán muchas sombras en la parte de los pies.

El tipo de pruebas que se realizarán son:

- **Misma posición.**

  -De frente
  -De perfil

- **Diferente posición**

  -De perfil

- **Movimiento exagerados**

  -Saltando

5.5.5.Análisis de los resultados

Los vídeos con los resultados obtenido pueden verse en la carpeta del DVD **Pruebas > 5.5 REALIZACION DEL CHROMA KEY.**

Para las pruebas que se realizaron el primer día, es decir, las que se grabaron en plano americano del sujeto y del primer plano se obtuvieron los siguientes resultados.

-Sujeto en la misma posición: **mismaposi_comp.flv**

-Sujeto cambiando de posición: **diferposi_comp.flv**

-Primer plano del sujeto: **primerplano_comp.flv**

Para estas pruebas se eligieron los vídeos que mejor valor de exposición dieron. En estos vídeos se compara el resultado de aplicar el efecto de chroma key al mismo vídeo. El vídeo se ha editado, dividiendo la imagen en dos partes. En la parte de la izquierda aparece un vídeo al que se le ha aplicado el efecto de Ultra key y se ha dejado el fondo negro y el de la derecha, es el mismo vídeo, con los mismos parámetros del efecto *Ultra key,* pero al que se le ha colocado un fondo con colores. Se aprecia en los vídeos, que los resultados son muy buenos, ya que en el peor caso, que son los vídeos con fondo negro, se ve un contorno verde alrededor del sujeto, pero al colocar un fondo, este contorno verde deja de apreciarse.

Los vídeos en los que el sujeto realiza los movimientos en la misma posición serán descartados para las grabaciones finales, ya que el movimiento es muy poco realista. Además, a la hora de incluir el personaje en flash, si se quiere que recorra el escenario habrá que añadir código para lograrlo. Esto además de complicar todo más, no mejoraría el movimiento poco real que se tiene, por lo que queda totalmente descartado

Los resultados obtenidos, para las pruebas del segundo día se muestran en los vídeos siguientes:

-Cuerpo entero del sujeto: **javin_pies_comp.flv**, **carol_pies_fondo.flv**, **carol_perfil_fondo.flv** y **javicarol_fondo.flv.**

En los vídeos que se eligieron de las grabaciones del segundo día de cuerpo entero se realizó lo mismo que para las explicadas anteriormente. Se aplicó el efecto Ultra key sobre la imagen y se comparó el resultado sobre un fondo negro y sobre un fondo con colores.

En estos vídeos lo que más interesaba era, el resultado de eliminar el fondo sobre la parte de los pies, ya que es la parte mas problemática por la gran aparición de sombras y cambios de luz que aparecen.

Los resultados que se obtuvieron fueron realmente buenos, ya que, para lograr eliminar el fondo de toda la imagen correctamente, tuvo que dejarse la zona de los pies con sombras. Esto en vez de perjudicial, resultó beneficioso, ya que ofrecía un efecto más realista en la imagen.

Los resultados fueron igual de buenos para todos los vídeos, tanto en los que el personaje no cambiaba de posición, como en los que sí e incluso para los que se hacían movimientos exagerados y por el mismo motivo que en las grabaciones anteriores, debido al efecto poco realista de los movimientos en la misma posición, se decidió que las grabaciones finales del personaje, se harían de cuerpo entero y cambiando de posición.

Capítulo 6: DISEÑO DE LA AVENTURA GRÁFICA

Una vez realizadas todas las pruebas, aplicaciones y grabaciones se comenzó a diseñar la aventura gráfica. Para ello se elaboró un guión artístico y un guión técnico.

En el guión técnico se cuenta la historia, se describen los diferentes escenarios que componen la aventura y los objetos y personajes que existen en cada uno. También se describe de forma muy simple los movimientos que puede hacer el protagonista y las acciones que podrán hacerse sobre cada objeto.

En el guión técnico, se describe lo mismo que en el artístico, pero de un forma más técnica y detallada. Para visualizarlo mejor, con cada explicación de las acciones que pueden realizarse sobre los objetos, se mostrará un esquema que detalla las acciones y sus resultados. Esto además, servirá de ayuda a la hora de programar. Lo mismo se hará con los movimientos del personaje, se han creado unos diagramas que muestran los movimientos en función de la posición inicial del personaje. Estos diagramas serán muy útiles a la hora de programar al personaje principal.

6.1 GUIÓN ARTÍSTICO

Esta aventura gráfica consta de siete escenarios en total, en los cuales existen diferentes objetos y personajes con los que se puede interactuar, que más adelante se detallarán. El protagonista, al que se le llamará a partir de ahora, Charli, tendrá libertad en todo momento, esto quiere decir, que podrá pasar de un escenario a otro aunque se haya dejado algún objeto sin coger o un personaje sin hablar. El jugador tendrá que pasar por los diferentes escenarios, buscando las pistas y recogiendo los objetos para cumplir el objetivo final del juego.

En este guión se explica brevemente la historia, se describen los objetos con los que se puede interactuar (círculo verde) y los personajes (marrón) que hay en cada escenario, además del recorrido que podrá hacer el personaje principal (rojo) y los lugares donde se situará (circulo rojo) para realizar las diferentes acciones como coger, hablar, etc.

El punto de partida de la aventura es el aulario, se mostrará una animación en la que Charli, llega y se dirige al tablón donde encuentra una nota extraña **(1)** que el protagonista coge. En ese momento se muestra una ampliación de la nota en la que se explica que ha ocurrido algo raro en la universidad, una invasión alienígena (no se contará todo de una manera explícita, ya que el jugador lo irá descubriendo a medida que avanza en el juego), que existe un objeto con el que puede ver la verdadera identidad de los personajes y que debe encontrar una serie de objetos para construir un artefacto que podrá usar para conseguir su objetivo, destruir a los aliens (se mostrará una silueta del artefacto como pista). Una vez ha leído la nota, se agrega al inventario y el protagonista avanza hasta el primer escenario donde empieza el juego.

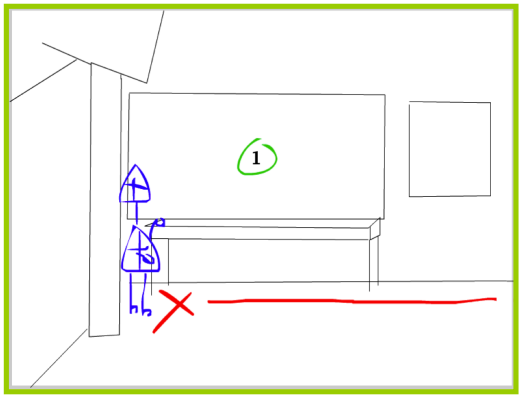


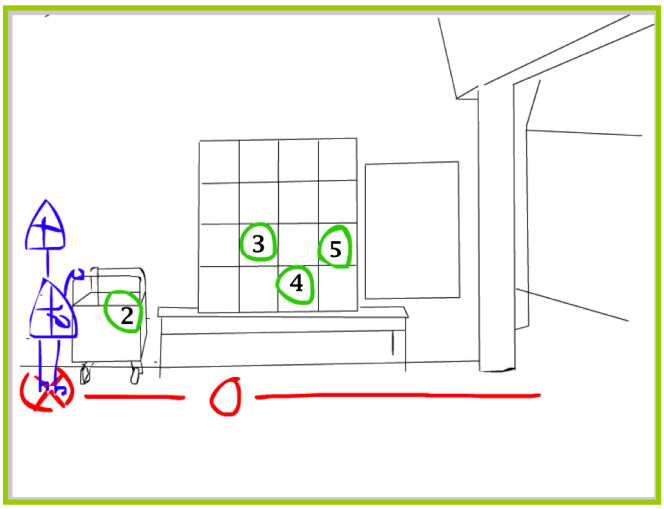
Figura 6.1.1. Boceto del Aulario 1

Una vez se ha pasado al primer escenario del juego, ya no se podrá volver a este punto en ningún momento del juego.

A continuación se describe por escenarios los objetos útiles que existen, los personajes y las acciones que pueden realizarse con los diferentes objetos y personajes.

➢ AULARIO

El primer escenario, donde comienza el juego, estarán las **taquillas,** donde habrá tres útiles, en una Charli tendrá que usar una **llave** para abrirla, donde estarán las **lentillas**, en otra, habrá que **dibujar** un símbolo, que se verá al colocarse las lentillas, donde estará el **monóculo** y en la última habrá que introducir una **clave**, donde encontraremos un **limón.** La **llave** se encuentra en el suelo al lado del carrito.



Objetos:
- -Llave **(2)**
- -Taquilla 1: lentillas **(3)**
- -Taquilla 2: monóculo **(4)**
- -Taquilla 3: limón **(5)**

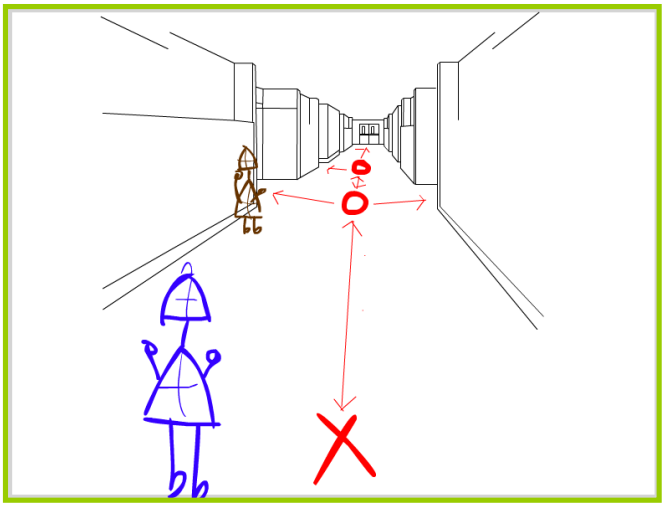
Gestos en **movimiento:**
- -De perfil derecho

Gestos quieto:
- -De perfil derecho mano

Figura 6.1.2. Boceto del Aulario

➢ PASILLO

Es el punto que une todos los escenarios, desde él se puede ir a cualquiera de ellos, no hay objetos, pero si **un personaje** con el que se puede interactuar,



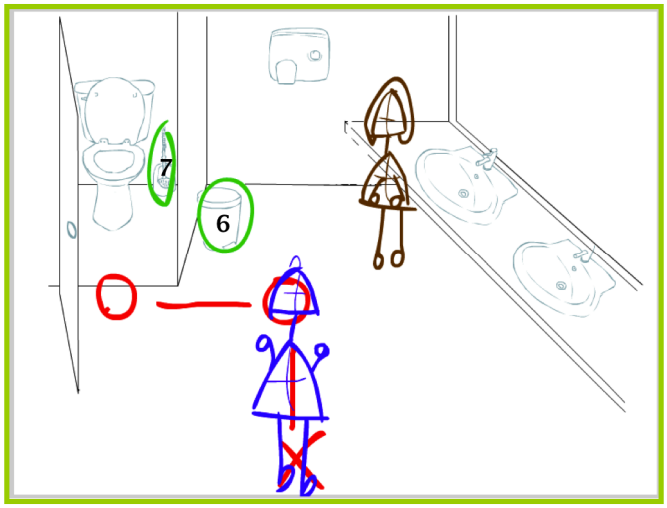
Actores:
- -Diego hablando por el móvil.

Gestos en movimiento:
- -De espaldas
- -De frente
- -De perfil derecho
- -De perfil izquierdo

Figura 6.1.3. Boceto del Pasillo

➢ BAÑO

En este escenario se encuentra la **escobilla** y el **secador roto**. El secador está dentro de la papelera, para cogerlo, antes se deberá abrir la papela, así se podrá ver el secador y cogerlo. En este escenario hay **un personaje** con el que interactuar, que además es **alienígena.**



Objetos:
-Escobilla de water **(7)**
-Secador en la papelera **(6)**

Actores:
-Eli pintándose las uñas

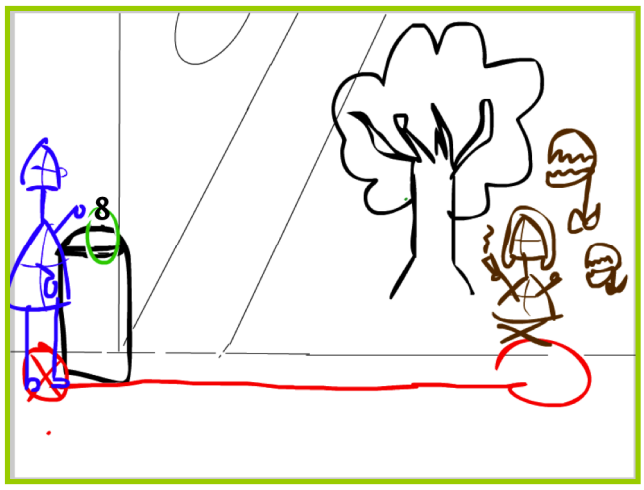
Gestos en movimiento:
-De espaldas
-De frente
-De perfil derecho
-De perfil izquierdo

Gestos quieto:
-De espaldas alargando el pie
-De espaldas agacharse

Figura 6.1.4 Boceto del baño

➢ EXTERIOR

En este escenario habrá un objeto que es la **pajita**, situada en la papelera y dos personajes. **Una persona** con la que se podrá interactuar y un **perro** que en realidad es un **alienígena.**



Objetos:
-Pajita **(8)**

Actores:
-Carol fumando

Gestos en movimiento:
-De perfil derecho
-De perfil izquierdo

Gestos quieto:

-De perfil derecho mano

Figura 6.1.5 Boceto del Exterior

➢ PLATÓ

En este escenario está el **ordenador con un tornillo suelto** y la **cinta americana**. Con el ordenador se deberá usar una herramienta para obtener el **tornillo**, creada a partir de dos objetos (un gato chino y una llave fija) que se encuentran en otro escenario y que previamente se han tenido que coger y combinar. También hay **un personaje** con el que se puede interactuar.

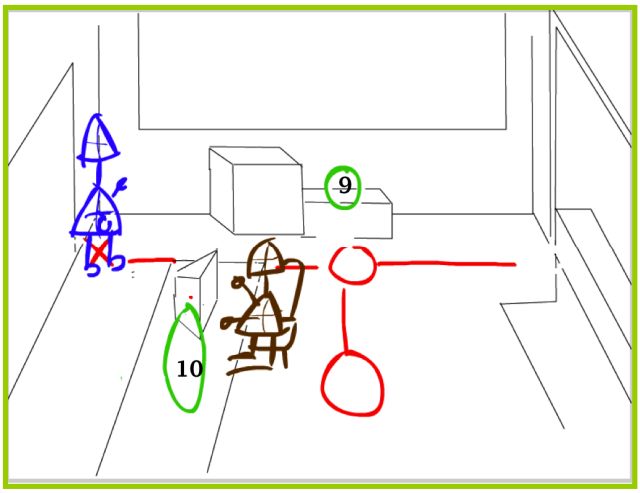


Figura 6.1.6. Boceto del plató

Objetos:
-Cinta americana **(9)**
- Ordenador **(10)**

Actores:
-Imanol escribiendo en el ordenador

Gestos en **movimiento**:
-De espaldas
-De frente
-De perfil derecho
-De perfil izquierdo

Gestos quieto:
-De perfil derecho mano
-De perfil izquierdo mano

➢ CHROMA

En el hay dos objetos, el **gato chino** y la **llave fija,** que podrán combinarse para formar un nuevo objeto. Este nuevo objeto se usa con el ordenador del plató para obtener el tornillo. También hay **dos personajes** con las que se puede interactuar, una de ellos es **alienígena.**

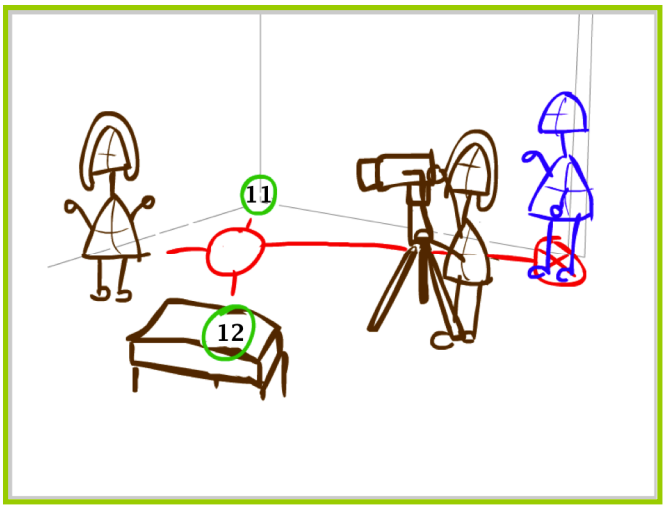


Figura 6.1.7 Boceto del Chroma

Objetos:
-Gato chino **(12)**
-Llave fija **(11)**

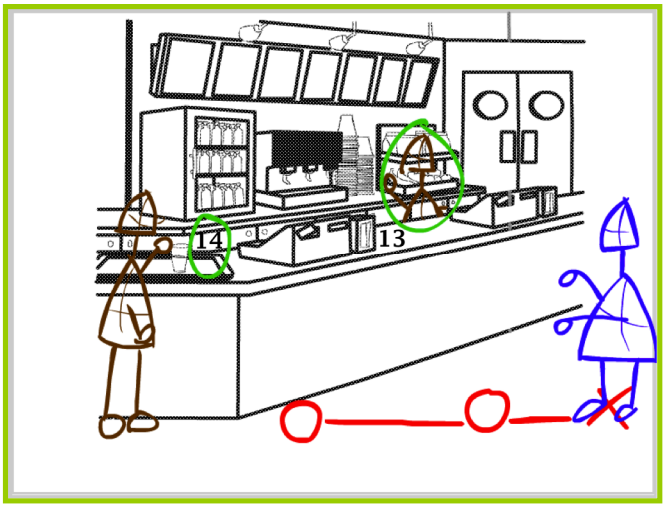
Actores:
-Aloha bailando
-Irene grabando

Gestos en movimiento:
-De perfil derecho
-De perfil izquierdo

Gestos quieto:
-De frente alargar la mano
-De espaldas agacharse

➢ CAFETERÍA

Aquí se puede coger un **embudo,** situado encima de la barra e interactuar con **dos personajes.** Al interactuar con la persona que hace de camarera aparecerán varias opciones para contestarle, dependiendo la opción que se elija, al final se obtendrá un botella de agua o no.



Objetos:
-Embudo **(14)**
-Botella de agua **(13)**

Actores:
-Ángel (Camarera)
-Javin bebiendo

Gestos en **movimiento:**
-De perfil derecho
-De perfil izquierdo

Gestos quieto:
-De perfil izquierdo mano

Figura 6.1.8 Boceto de la cafetería

Una vez se hayan conseguido todos los objetos, se podrán ir combinado en el orden correcto para obtener el arma (también se pueden ir combinado los objetos, a medida que se van cogiendo a lo largo del juego).

Cuando se han combinado todos los objetos, se obtendrá un arma, que deberá usarse contra los aliens (se habrá tenido que usar las lentillas para poder ver a los aliens). Esta arma no producirá ningún efecto sobre estos. Al usar el arma sobre el alien que se encuentra en el baño, le dará una pista para obtener el último objeto. Esta pista será la clave que habrá que utilizar en la taquilla restante donde se encuentra el limón, al combinar el limón con la pistola, se obtendrá el objeto final, que habrá que combinar con la pistola y que, al usar con los aliens, hará que desaparezcan. Cuando se haya disparado a todos los aliens, el juego terminará con una animación final.

6.2. GUIÓN TÉCNICO

En este guión se explica lo mismo que en el guión artístico pero de una forma más amplia y técnica. Se muestran las posiciones en las que el personaje ejecutará las acciones y los gestos que podrá hacer, los objetos activos que podrá coger, usar o combinar y los personajes con los que podrá interactuar.

Habrá **cuatro tipos de objetos**. Los que se podrán **almacenar** en el inventario, los que se podrán **combinar** con otros objetos, los que se podrán **usar** con elementos del escenario, y con los que podrá **hablar**, es decir los personajes que irán apareciendo con los que podrá dialogar.

ALMACENAR: Hacer click sobre un objeto del escenario y que se almacene en el inventario.

COMBINAR: Arrastrar un objeto del inventario sobre otro objeto del inventario y crear un nuevo objeto.

USAR: Arrastrar un objeto del inventario sobre un objeto del escenario para que ocurra algo.

HABLAR: Hacer click sobre un personaje y que aparecer bocadillos de conversación.

En el escenario habrá objetos activos, que al hacer click sobre ellos, el protagonista avanzará hasta una posición determinada para poder interactuar con ellos, estos sitios están marcados con una **p+número.** También se podrá acceder a estas posiciones en el escenario, sin interactuar con un objeto o personaje.

A continuación se explicará escenario por escenario todo lo comentado:

El punto de partida es una animación en la que el jugador, no tendrá que hacer nada. Se mostrará como llega el personaje al aulario, va hacia el tablón y encuentra la nota **(1)**, la lee y la guarda en el inventario, una vez ocurre esto, se pasa directamente al primer escenario del juego.

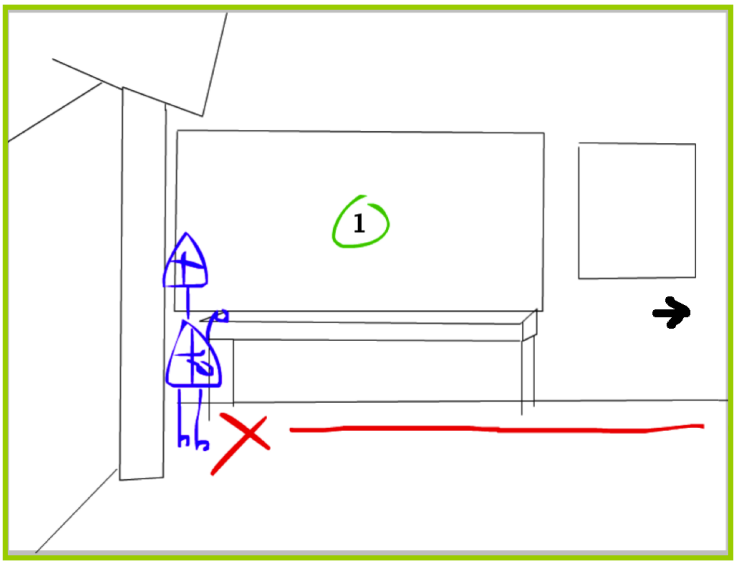


Figura 6.2.1 Boceto del Aulario

AULARIO

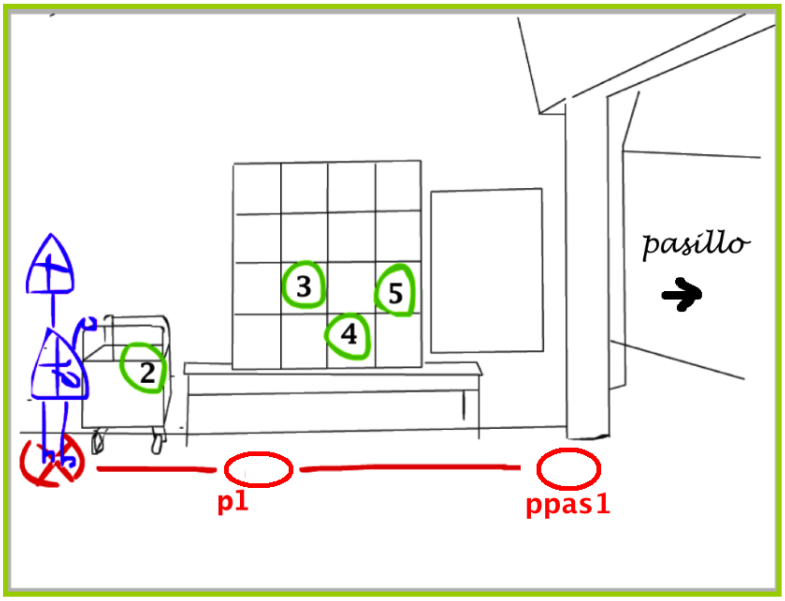


Figura 6.2.2. Boceto del Aulario con posiciones

Una vez en el aulario, el protagonista no podrá volver al punto anterior, pero si podrá ir a cualquier otro escenario.

Movimientos

Al hacer click en cualquiera de las taquillas **(3), (4)** o **(5),** el protagonista avanzará hasta el circulo rojo, posición **p1.**

Al hacer click en el **llave** el protagonista se moverá hasta el punto inicial marcado con una **x.**

Al hacer click en **pasillo** se moverá hasta otro escenario, el pasillo.

Los movimientos que se realizarán en el aulario dependiendo de donde se encuentre el protagonista vienen detallados en el *diagrama 6.1*:

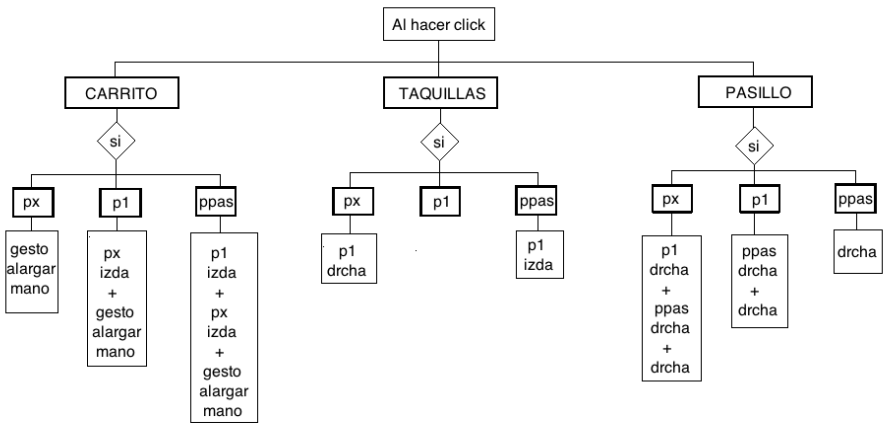


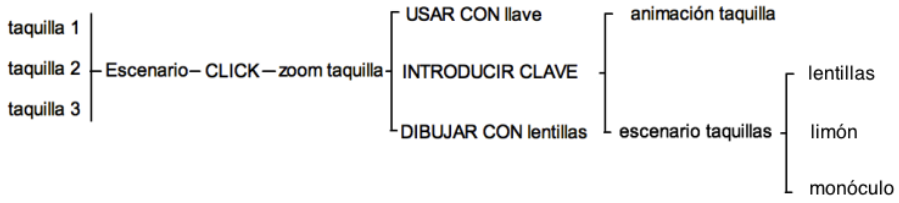
Diagrama 6.1. Aulario

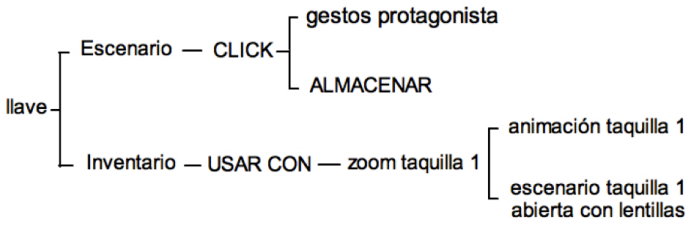
Objetos:

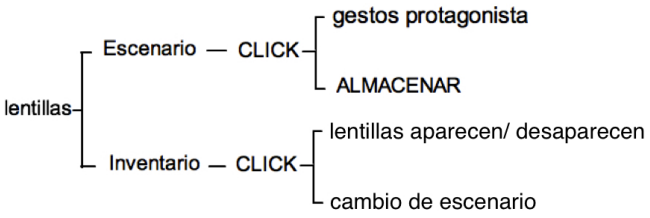
En este escenario, habrá una llave **(2)** que al combinarla con la taquilla 1 **(3),** aparecerá una animación de una puerta de taquilla que se abre y dentro unas lentillas, que se podrán usar en cualquier momento.

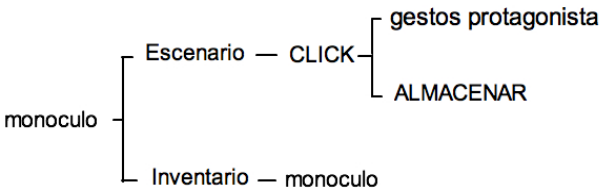
La taquilla 2 se abre haciendo un dibujo, el protagonista deberá ponerse las lentillas y así vera la silueta del dibujo que tendrá que hacer, al hacerlo bien, se abrirá la taquilla y se podrá coger el monóculo.

Las acciones que se pueden realizar con cada objeto vienen detalladas en los esquemas siguientes:









Personajes:

En este escenario no habrá ningún personaje.

➢ PASILLO

Este escenario es el que une todos los demás, desde aquí se puede acceder a todos ellos, y al salir de cada escenario se vuelve a este.

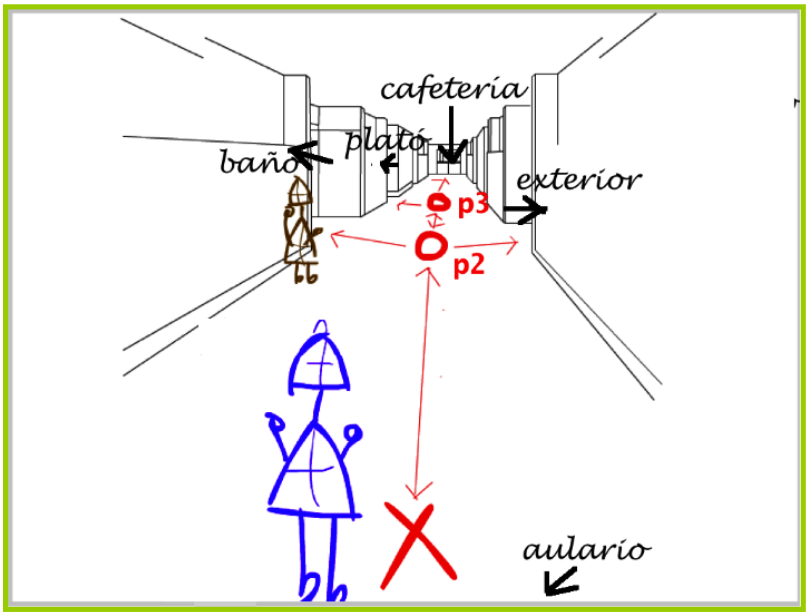


Figura 6.2.3. Boceto del Pasillo con posiciones

Movimientos:

Al hacer click en baño el personaje avanzará hasta la posición **p2** y luego girará hacia la izquierda

Al hacer click en exterior el personaje avanzará hasta la posición **p2** y luego girará a la derecha

Al hacer click en plató, el personaje pasará por **p2** hasta llegar a **p3** y luego girará a la izquierda

Al hacer click en cafetería el personaje pasará por **p2** y **p3** y seguirá avanzando hasta la puerta del fondo.

Al hacer click sobre el personaje, el protagonista avanzará hasta la posición **p2**

Desde todos los escenarios, se puede ir a los demás pasando por las posiciones **p2** y **p3** y girando hacia el lugar correspondiente.

Los movimientos que se realizarán en el pasillo dependiendo de donde se encuentre el protagonista vienen detallados en el *diagrama 6.2*

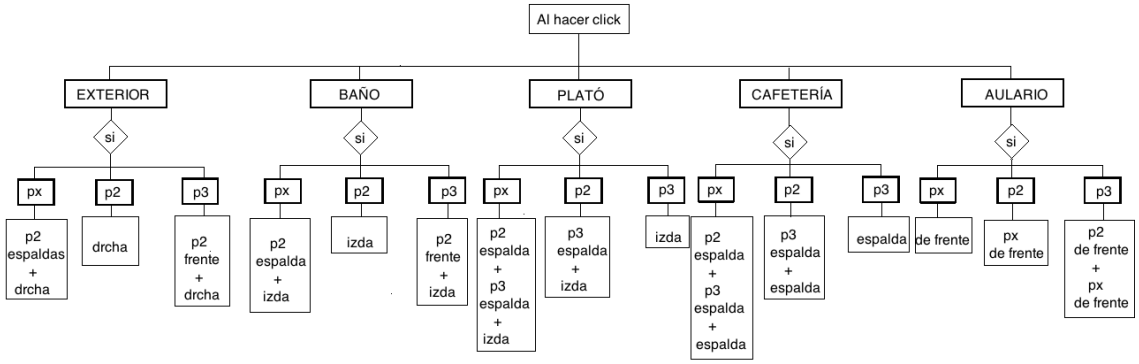
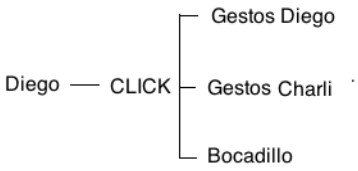


Diagrama 6.2. Pasillo

Personaje:

En este escenario estará Diego hablando con el móvil. Al hacer click sobre él, los gestos de Diego cambiarán y aparecerá un bocadillo. En el esquema siguiente se detalla:



➢ BAÑO

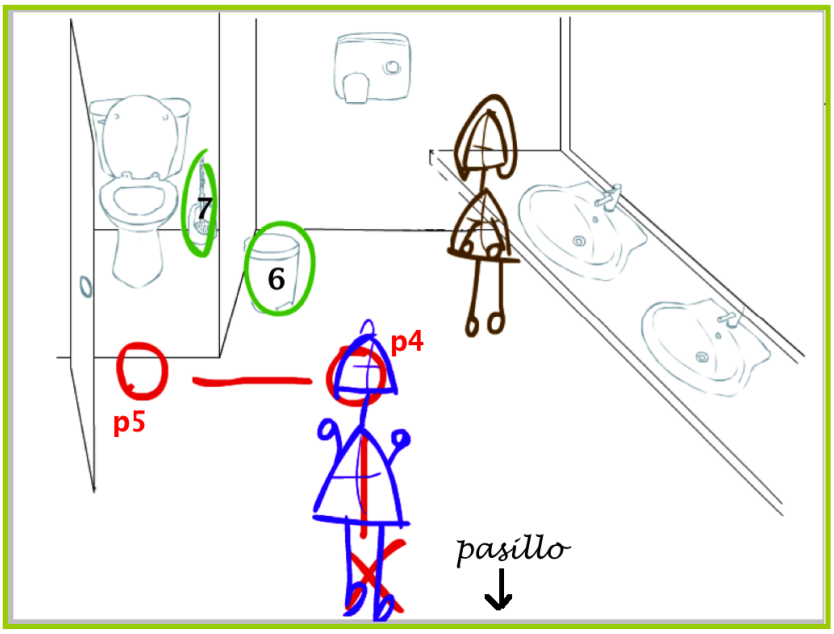


Figura 6.2.4 Boceto del baño con posiciones

Movimientos:

Al hacer click en la papelera o en el personaje, el protagonista avanzará hasta la posición **p4.**

Al hacer click en la escobilla el personaje primero pasará por la posición **p4** y luego **p5**. Si está en la posición **p4** el personaje solo girará hacia la izquierda hasta la posición **p5**.

Al hacer click en pasillo, volveremos por el recorrido, hasta el pasillo.

Los movimientos que se realizarán en el baño dependiendo de donde se encuentre el protagonista vienen detallados en el *diagrama 6.3.*

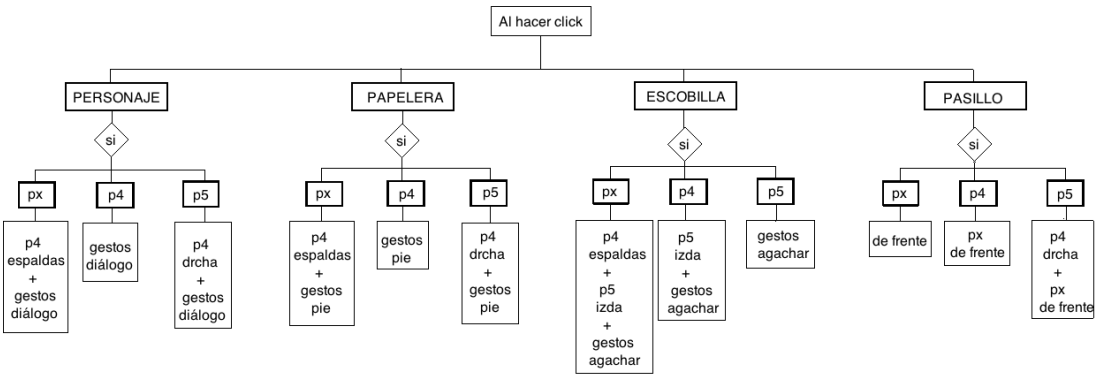
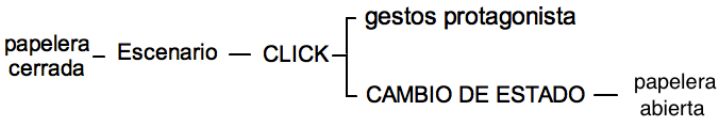


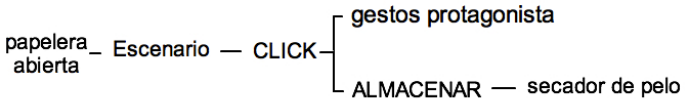
Diagrama 6.3. Baño

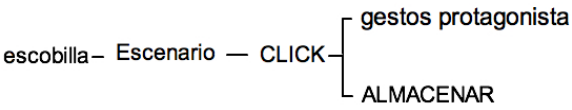
Objetos:

En este escenario se puede interactuar con la papelera **(6)** que al hacer click sobre ella, cambiará de estado haciendo que esté abierta y al volver a hacer click podremos coger el secador. También se podrá coger la escobilla **(7).**

Las acciones que se pueden realizar con cada objeto vienen detalladas en los esquemas siguientes:





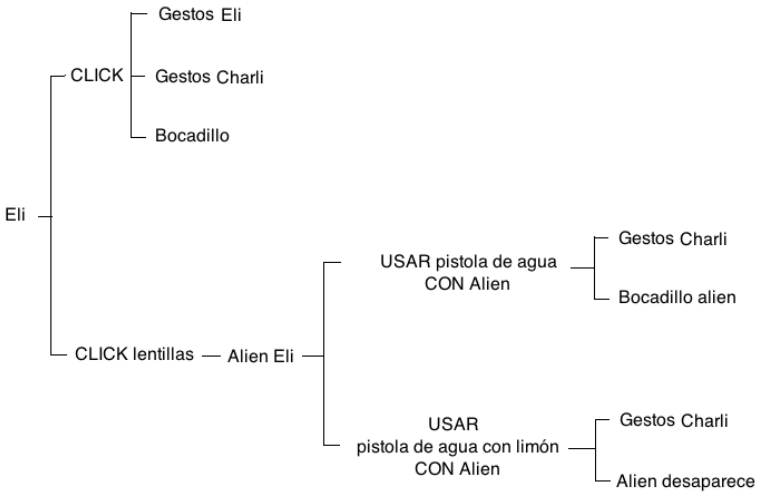


Personajes:

En este escenario estará Eli pintándose las uñas, este personaje además es un alien.

Con los personajes que son aliens, se puede interactuar de tres formas diferentes. Al hacer click sobre el personaje, aparecerán los bocadillos de conversación y los gestos del personaje y del protagonista cambiarán. Cuando el personaje se ve como alien, al usar la pistola de agua con él (arrastrándola y soltándola encima de él), aparecerán otros bocadillos de conversación diferentes a los anteriores y por último, al usar la pistola de agua con limón, el alien desaparecerá.

En el esquema siguiente se aprecia mejor cada acción:



➢ EXTERIOR

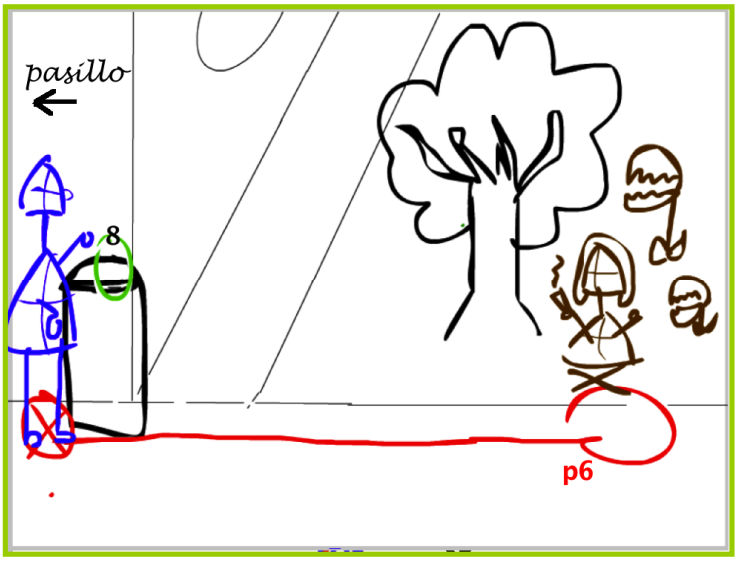


Figura 6.2.5. Boceto del Exterior con posiciones

Movimientos:

Al hacer click en la papelera o en el personaje, el protagonista avanzará hasta la posición **inicial.**

Al hacer click en el personaje o las plantas, el protagonista avanzará hasta la posición **p6**

Al hacer click en **pasillo**, volveremos por el recorrido, hasta el pasillo.

Los movimientos que se realizarán en el exterior dependiendo de donde se encuentre el protagonista vienen detallados en el *diagrama 6.4*:

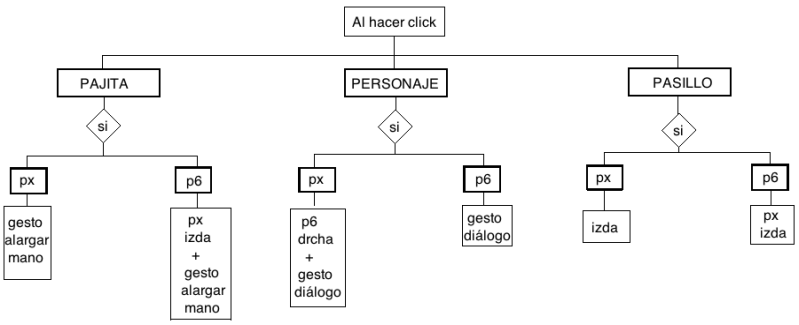
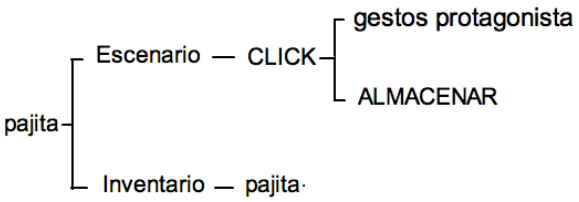


Diagrama 6.4. Exterior

Objetos:

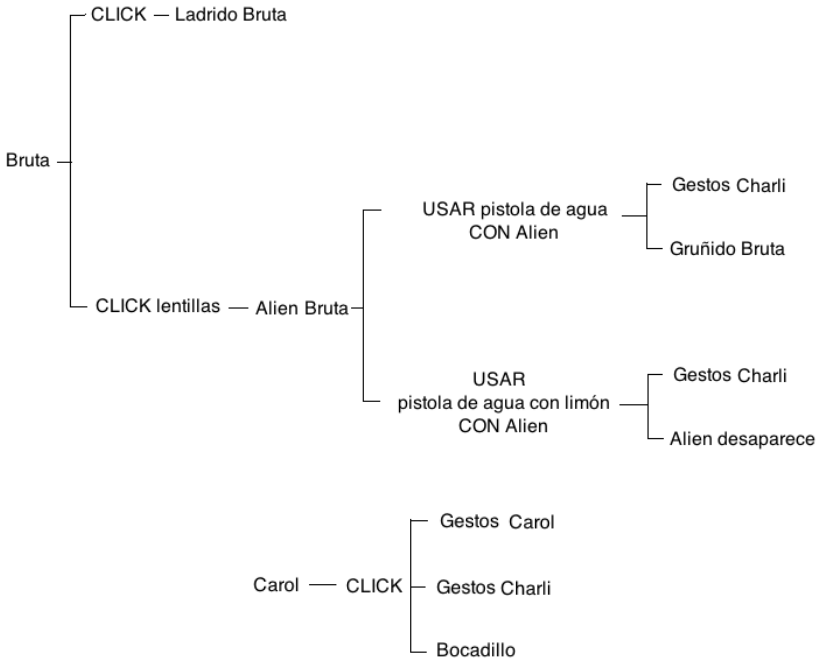
En este escenario podremos coger la pajita **(8)** que está en la papelera.

Las acciones que se pueden realizar con cada objeto vienen detalladas en los esquemas siguientes:



Personajes:

Aquí estará Carol fumando y un perrito llamado Bruta que será un alien. Las acciones que pueden hacerse son:



- PLATO

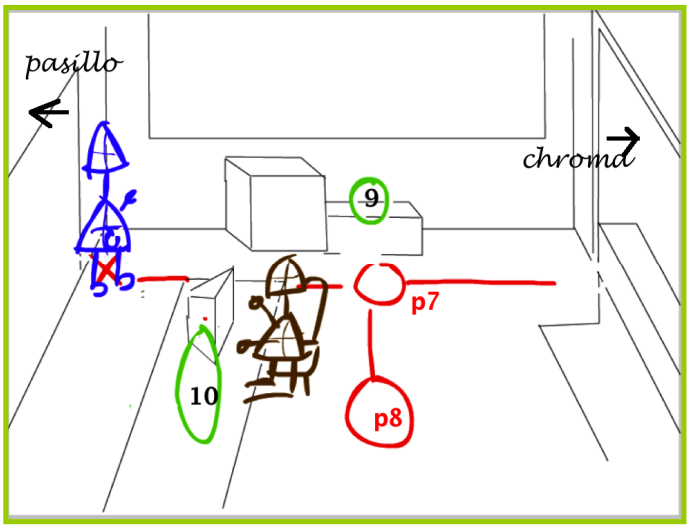


Figura 6.2.6. Boceto del plató con posiciones

Movimientos:

Al hacer click en la cinta americana, el protagonista avanzará hasta la posición **p7.**

Al hacer click en el ordenador, el protagonista avanzará primero hasta la posición **p7** y después girara hasta la posición **p8.**

Al hacer click en **chroma**, el protagonista avanzará hasta el siguiente escenario chroma

Al hacer click en **pasillo**, volveremos por el recorrido, hasta el pasillo.

Los movimientos que se realizarán en el plató dependiendo de donde se encuentre el protagonista vienen detallados en el *diagrama 6.5*:

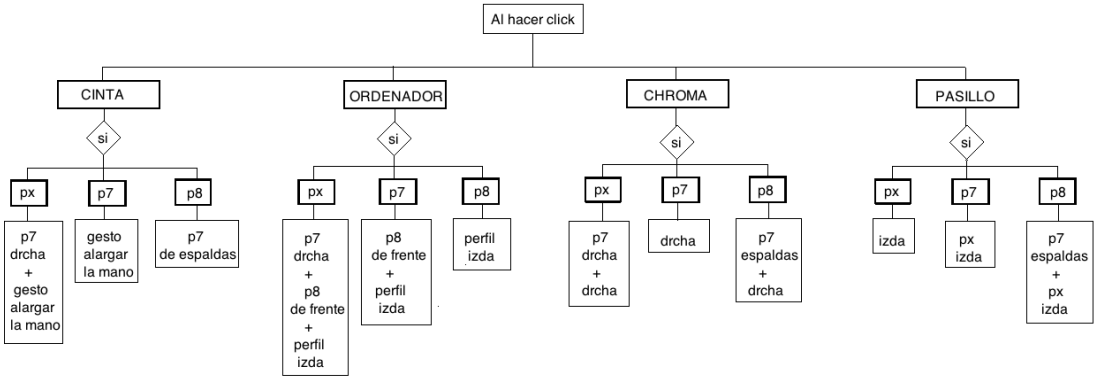
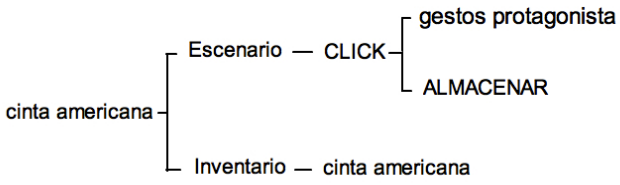


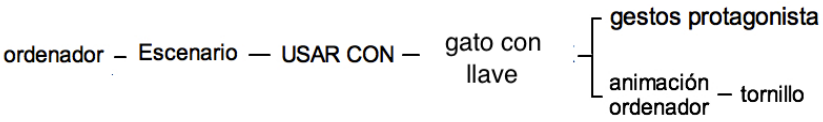
Diagrama 6.5. Plató

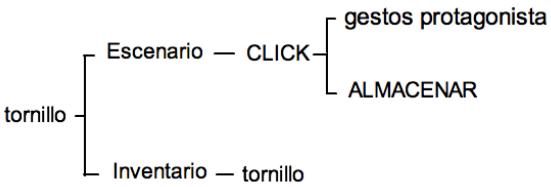
Objetos:

En este escenario se puede coger la cinta americana **(9)** e interactuar con el ordenador **(10),** para conseguir el tornillo, para ello, antes se tendrá que combinar el gato con la llave, y luego combinar esto con el ordenador, aparecerá una animación del ordenador destrozado y un tornillo al lado.

Las acciones que se pueden realizar con cada objeto vienen detalladas en los esquemas siguientes:

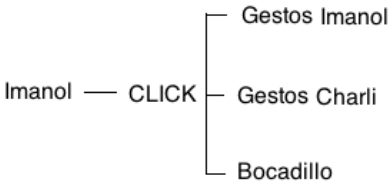






Personajes:

En este escenario, esta Imanol en el ordenador. Las acciones que puede hacer al hacer click son:



➢ CHROMA

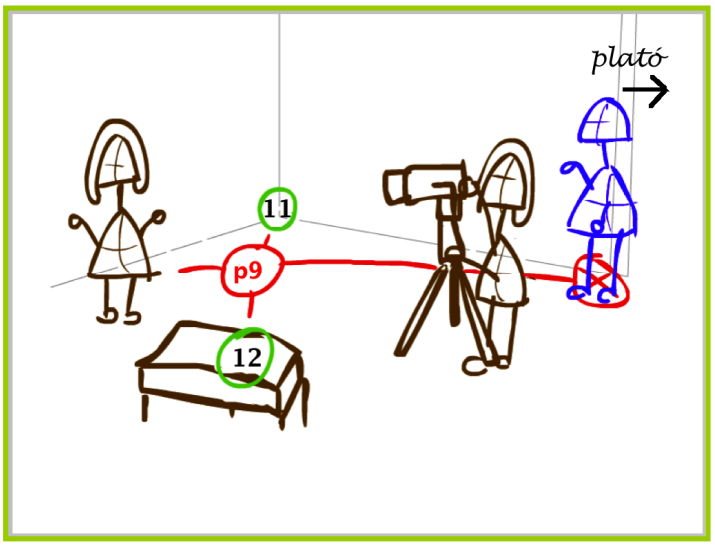


Figura 6.2.7. Boceto del Chroma con posiciones

Movimientos:

Al hacer click en la cortina, el protagonista avanzará hasta la posición **p9** y se quedará de espaldas.

Al hacer click en gato, el protagonista avanzará primero hasta la posición **p9** y se quedará de frente

Al hacer click en el personaje con la cámara se quedara en la posición inicial

Al hacer click en el personaje, el protagonista avanzará hasta la posición **p9** y se quedará de perfil

Al hacer click en **plató**, volveremos por el recorrido, hasta el plató.

Al hacer click en **pasillo**, volveremos por el recorrido, hasta el pasillo.

Los movimientos que se realizarán en el chroma dependiendo de donde se encuentre el protagonista vienen detallados en el *diagrama 6.6.*

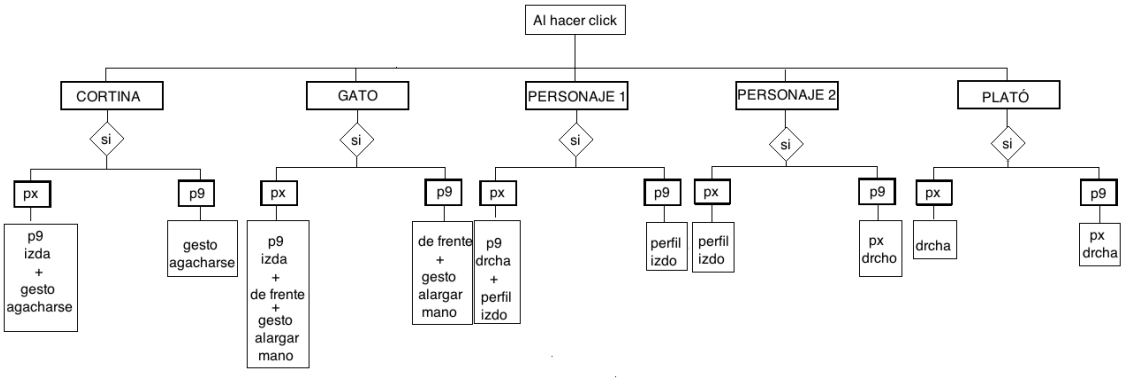
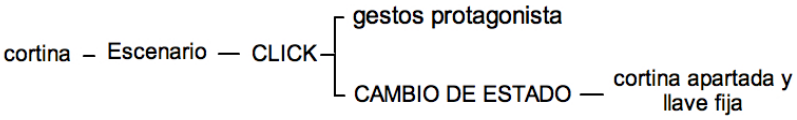


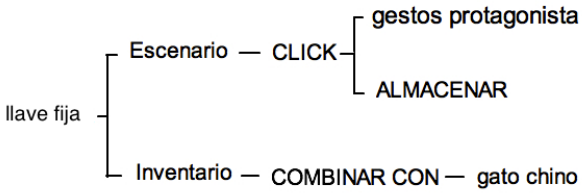
Diagrama 6.6. Chroma

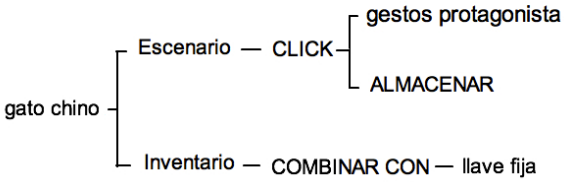
Objetos:

En este escenario se puede coger la llave fija **(11)** y el gato chino que está encima de la mesa **(12).**

Las acciones que se pueden realizar con cada objeto vienen detalladas en los esquemas siguientes:

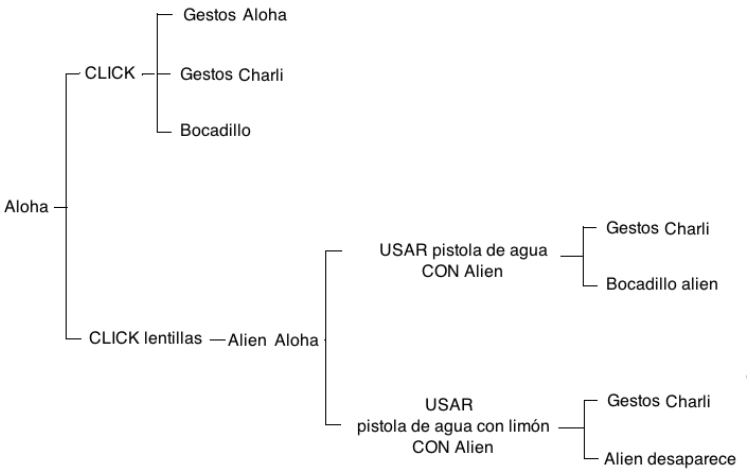


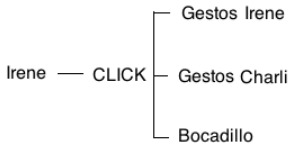




Personajes:

En este escenario, estarán Irene grabando y Aloha bailando, esta será un alien también. Las acciones que pueden hacer cada una son:





➢ CAFETERÍA

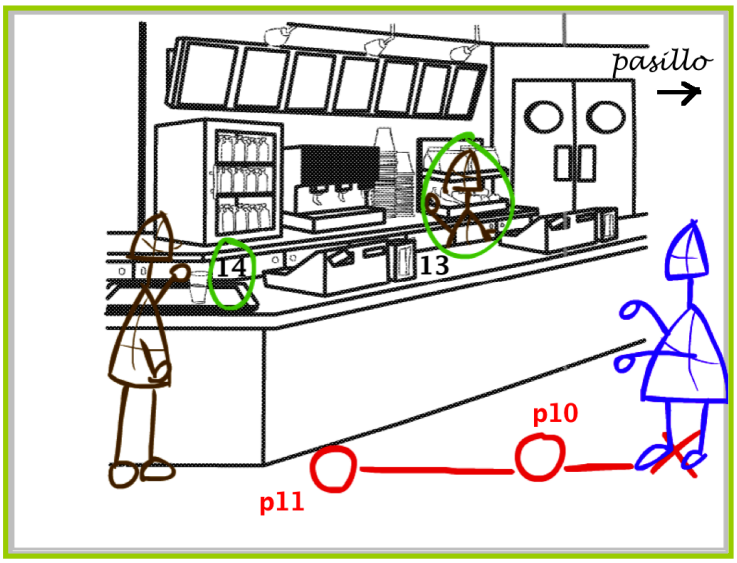


Figura 6.2.8. Boceto de la cafetería con posiciones

Movimientos:

Al hacer click en la camarera, el protagonista avanzará hasta la posición **p10.**

Al hacer click en el personaje o en el embudo, el protagonista avanzará hasta la posición **p11**

Al hacer click en **pasillo**, volveremos por el recorrido, hasta el pasillo.

Los movimientos que se realizarán en la cafetería dependiendo de donde se encuentre el protagonista vienen detallados en el *diagrama 6.7.:*

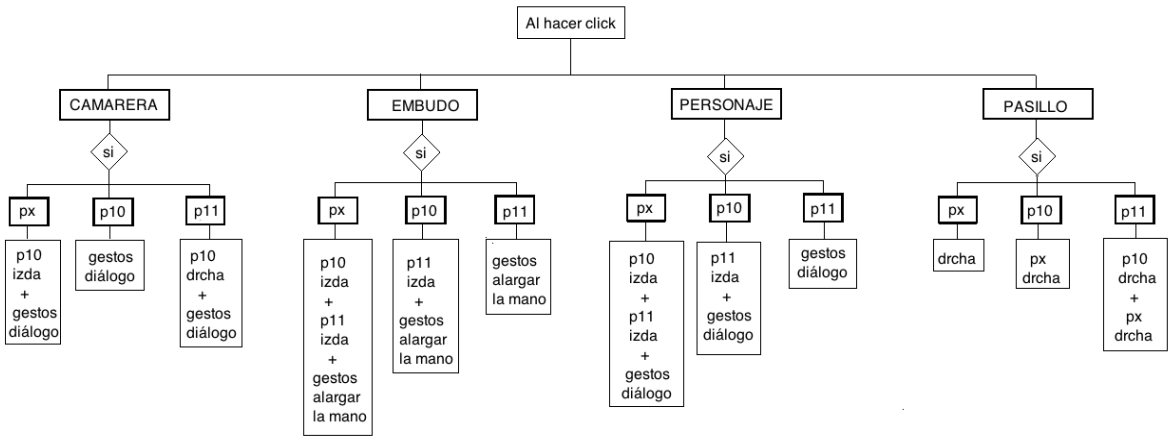
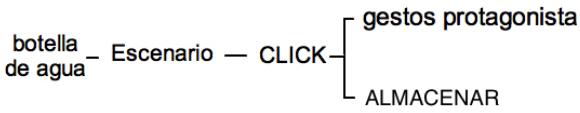


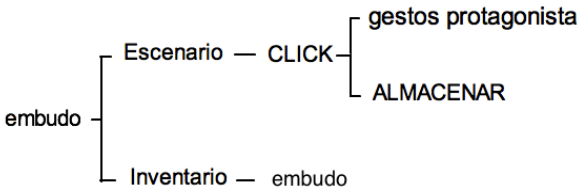
Diagrama 6.7. Cafetería

Objetos:

En este escenario se puede coger el embudo **(14)** e interactuar con la camarera, hablaremos con ella y dependiendo de lo que le digamos (habrá varias opciones) nos dará una botella de agua **(13).**

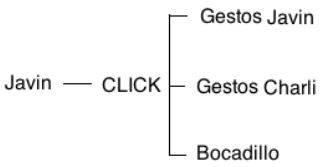
Las acciones que se pueden realizar con cada objeto vienen detalladas en los esquemas siguientes:





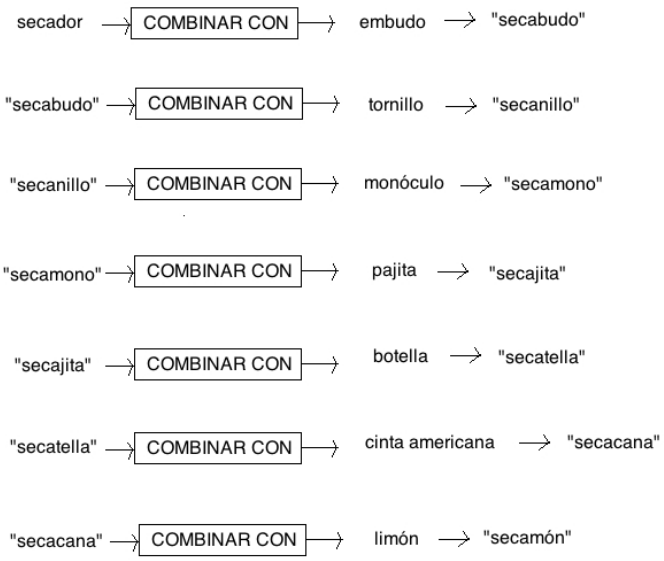
Personajes:

En este escenario, estará Javin en la barra bebiendo una cerveza, que también es alien y Ángel dentro de la barra, que será la camarera. Las acciones de Ángel diferentes, ya que al hacer click sobre él, se tendrán varias opciones para elegir y dependiendo de la que se elija, aparecerán unos bocadillos de conversación diferentes y los gestos de los personajes también serán diferentes para cada opción. Estas son las acciones:



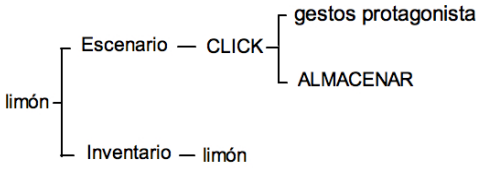
Una vez se tienen todos los objetos se podrá montar el arma, habrá que montarla en un orden determinado. Cuando se tenga el arma montada se almacenará en el inventario.

El arma solo puede usarse con los aliens, así que habrá que usar las lentillas, para verlos y arrastrar la pistola hacia ellos, para que ocurra algo. El orden para montar la pistola será:



Cuando se prueba a usar la pistola de agua con los aliens, aparecen bocadillos de conversación. El alien del baño, da la pista para conseguir el último objeto.

En el aulario está el último objeto que se conseguirá al introducir la clave que ha dado el alien del baño. Dentro de la taquilla está el limón que habrá que coger.



El limón se combina con la pistola para crear la última fase de esta. Al usarla con los alien, hará que estos desaparezcan. Existirá un contador desde 0, que irá sumando uno cada vez que se dispare a un alien, cuando llegue a cuatro (los aliens totales) aparecerá una animación final y el juego habrá terminado.

6.3. GRABACIÓN FINAL DEL PERSONAJE

Para finalizar la parte del diseño de la aventura gráfica, se elaboró un guión para las grabaciones de la persona que haría el papel protagonista de la aventura. En el se especifica el esquema de iluminación, los parámetros que son aconsejables utilizar en función de los resultados obtenidos en las grabaciones anteriores y los movimientos que tendrá que hacer el actor, a parte de lo de caminar, que son necesarios .

Una vez realizadas las pruebas de chroma, se decidió grabar al personaje de cuerpo entero, ya que los resultados de estas pruebas fueron muy buenos. El encuadre que se realizará del personaje será como el siguiente:

Figura 6.2.9. Captura de las pruebas realizadas de cuerpo entero

Se usará el esquema de iluminación y parámetros de la cámara que se utilizaron en los vídeos elegidos para el análisis de las pruebas de chroma de cuerpo entero.

En estas pruebas se colocará una cartulina verde en el suelo para que el personaje pueda dar un paso al frente.

Esquema de iluminación:

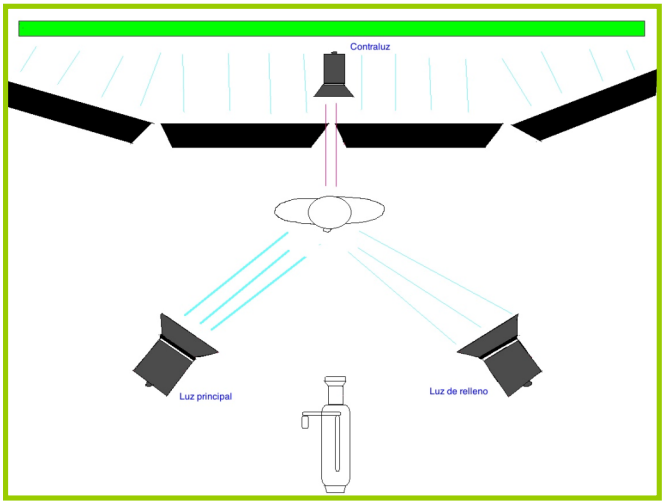


Figura 6.2.10. Iluminación a tres puntos

Parámetros de cámara:

Los parámetros de cámara con los que se obtuvieron los mejores resultados en las pruebas de cuerpo entero, fueron los siguientes:

-Diafragma 2,6
-Shutter 1/125
-Ganancia -3dB

Por lo tanto, manteniendo el mismo esquema de iluminación de las anteriores pruebas, se usarán esos valores o los más próximos para asegurar el mejor resultado.

Se grabará al sujeto realizando diferentes movimientos y gestos a cuerdo con los que se necesitarán para la aventura gráfica, son los siguientes:

6.3.1.- En el mismo sitio caminando:

-En el mismo sitio haciendo que camina hacia delante
-En el mismo sitio haciendo que camina caminando hacia detrás
-En el mismo sitio haciendo que camina caminando hacia la izquierda
-En el mismo sitio haciendo que camina caminando hacia la derecha

6.3.2.- Dando un paso:

-En el mismo sitio caminando dando un paso hacia delante
-En el mismo sitio caminando dando un paso hacia detrás
-En el mismo sitio caminando dando un paso hacia la izquierda
-En el mismo sitio caminando dando un paso hacia la derecha

6.3.3.- En el mismo sitio parado:

-Parado de frente
-Parado de espadas
-Parado de perfil izquierdo
-Parado de perfil derecho

6.3.4.- Haciendo gestos:

-Parado de perfil derecho alargando la mano
-Parado de perfil izquierdo alargando la mano
-Parado de frente alargando la mano
-Parado de espaldas, agacharse y alargar la mano para coger algo
-Parado de perfil y agacharse para coger algo
-Parado de espaldas y alargar el pie (pisar la papelera)

6.3.5.- Gestos disparar pistola:

-De frente disparando pistola
-De espaldas disparando pistola
-De perfil izquierdo disparando pistola
-De perfil derecho disparando pistola

6.3.6.- Gestos de aburrimiento:

-Parado de frente haciendo gestos de aburrimiento
-Parado de espaldas haciendo gestos de aburrimiento
-Parado de perfil derecho haciendo gestos de aburrimiento
-Parado de perfil izquierdo haciendo gestos de aburrimiento

6.3.7.- Gestos diálogo con personajes:

-Parado de perfil derecho haciendo gestos con las manos y hablando.
-Parado de perfil izquierdo haciendo gestos con las manos y hablando.

6.3.8.- Primer plano

-De espaldas, un poco girado (se ve parte de la cara)

Capítulo 7: DESARROLLO DE LA AVENTURA GRÁFICA

En esta última parte se explicará la estructura que posee el juego en base a la programación, así como las funciones más importantes que son las que se refieren a la interactividad de los personajes, los objetos y la forma en la que estos mantienen el estado que les corresponde después de haber realizado una acción determinada.

Para la parte de las funciones se incluirá una descripción de lo que hace la función, un diagrama de bloques si es necesario, ya que muchas de las funciones, al estar basadas en las aplicaciones mostradas en el capítulo 5, tendrán la misma estructura. Por último se hará una explicación más detallada de las partes del código, mostrando el propio código con los comentarios oportunos.

7.1 ESTRUCTURA

El juego esta formado por tres SWF:

- El primero es una pequeña animación, que posee dos botones. Con uno de ellos se accede a las instrucciones del juego y con el otro al propio juego.

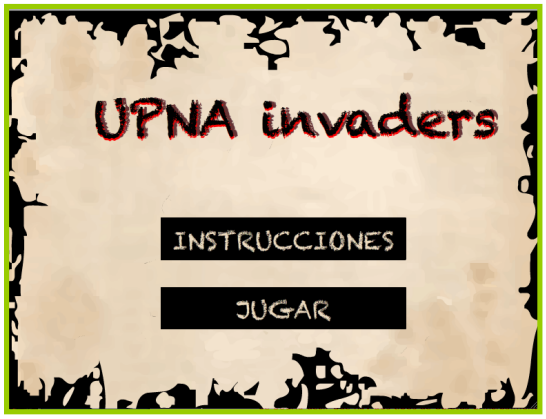


Figura 7.1 Menú principal

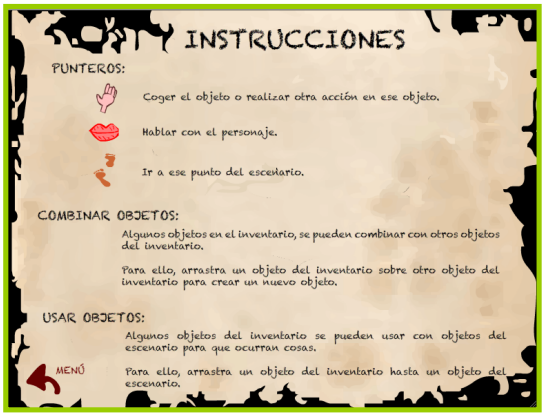


Figura 7.2 Instrucciones

- El segundo SWF es el juego, que es cargado (más tarde se explicará como) cuando se pulsa el botón de jugar, después de una pequeña animación.

- El tercero es una animación que se muestra cuando se ha finalizado el juego.

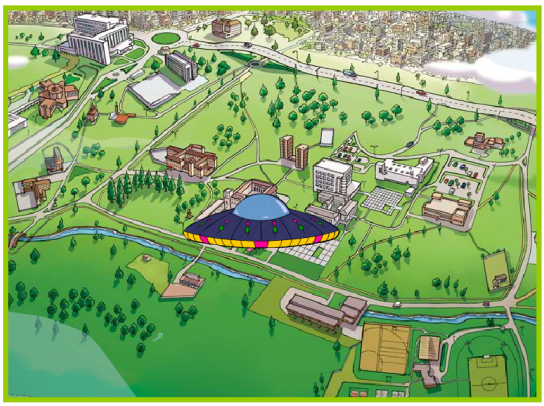


Figura 7.3 Animación final

El primer y último SWF son sencillas animaciones. Donde se encuentra toda la programación y el objetivo de este proyecto es en el segundo SWF. A continuación se explicará la estructura que posee el juego

7.1.1. Organización del código

La estructura del juego está dividida por los escenarios que lo componen. En total hay siete escenarios. Cada escenario esta situado en un fotograma diferente. Dentro de cada fotograma la programación se estructura de la siguiente forma.

- Se describen los objetos que son inicialmente visibles o no.

- Se asigna a cada objeto un texto mediante la propiedad text.

- Llamada a la función **Objetos+Escenario()**. Esta función se encarga de repasar todas las variables que afectan a los objetos que se encuentran en ese escenario para que estén en el estado que les corresponde. Más adelante se hablará de los tipos de variables que se ha creado para mantener el control de los objetos.

- A continuación se llaman a las funciones referentes a los objetos que se encuentran en ese escenario.

- La última parte es la que se refiere a la programación de los movimientos del personaje.

Esta sería la estructura básica de cada fotograma.

En el primer fotograma, no hay ningún escenario. En el se han colocado los objetos correspondientes al inventario para inicializarlos como no visibles y asignarles un texto.

Las definiciones de las distintas funciones se encuentran en los fotogramas que no pertenecen a los escenarios.

7.1.2. Distribución de objetos

Se podría decir que hay dos tipos de objetos principalmente. Los **objetos del escenario** y los **objetos del inventario**. Todos los objetos del juego están en el escenario principal. No se ha creado ningún contenedor especifico para cada tipo de objeto ya que se pretende que todas los objetos del inventario puedan ser arrastrados en cualquier momento para intentar interactuar con el resto de objetos, ya sean del escenario o del inventario. Lo que se pretende es que haya una gran libertad en el juego y que no se le de ninguna pista al el jugador.

La colocación de los objetos en los fotogramas cambia dependiendo si son de un tipo o de otro. Los objetos que se encuentran en el escenario y que pueden ser almacenados, estarán en una capa que abarca los fotogramas del escenario que les corresponde. Sin embargo los objetos de inventario se encuentran en una capa que abarca todos los fotogramas, para que puedan ser visualizados en todo momento.

En la *figura 7.4.* se aprecia mejor. La capa nombrada como inventario, contiene a todos los objetos del inventario y abarca todos los fotogramas. Sin embargo la capa objetos esta dividida según los escenarios

Figura 7.4 Distribución de los objetos por capas

Hay objetos que se encuentran en toda la línea de tiempo, pero que no pertenecen al inventario, como por ejemplo, la taquilla. Los objetos del escenario que interactúen con objetos del inventario, deben estar colocados en los mismos fotogramas. Sino ocurrirán errores al intentar arrastrar el objeto en un escenario que no se encuentre el objeto con el que debe interactuar, porque ActionScript no podrá localizar ese objeto.

7.1.3. Variables

Durante el juego se podrán realizar diversas acciones que harán que los objetos cambien de estado, por ejemplo, coger, disparar, cambiar escenario...

Para cada acción que se realice en un objeto, habrá una variable que almacene sus estados.

Por ejemplo, existe un objeto llamado *embudo*, que tiene asociado la variable, llamada tengoembudo, esa variable inicialmente tiene el valor **false**, que corresponde con el embudo del escenario visible y el del inventario invisible. Cuando el jugador hace click sobre el embudo, se ejecutan las funciones que llaman a los cuepoints necesarios para realizar los gestos de Charli. Cuando se detecta un cuepoint determinado, llamado "almacenarembudo", la variable cambia a **true**. Esta variable queda almacenada. El jugador puede abandonar el escenario, pero más tarde puede volver. Por este motivo se creo la función **Objetos+Escenario()**, que lee todas las variables asociadas a los objetos y dependiendo de su valor, colocará en un estado u otro a los objetos que correspondan.

Los tipos de variables que se han creado son:

- **tengo+objeto=** Puede tomar los valores true o false. Indica si un objeto ha sido almacenado o no.
  - false: objeto no almacenado.
  - true: objeto almacenado.

- **escenariocambiado**= Puede tomar los valores true o false. Indica si al escenario se le ha aplicado un filtro.
  -false: O se ha aplicado el filtro al escenario.
  -true: Se ha aplicado el filtro al escenario.

- **alien+nombre+disparado**= Puede tomar los valores true o false si se ha usado al pistola contra el alien y este ha desaparecido.
  -false: no se ha usado la pistola con el (permanece visible).
  -true: se ha usado la pistola con (permanece invisible).

- **tengolentillas**= Puede tomar los valores true o false si se han almacenado las lentillas.
  -false: no se han almacenado las lentillas.
  -true: se han almacenado las lentillas.

- **papeleraabierta**= Puede tomar los valores 0, 1 y 2. Indica si la papelera está abierta y en el caso de que si, si se ha almacenado el secador o no.
  -0: la papelera está cerrada por lo que no se visualiza el secador.
  -1: la papelera esta abierta, se visualiza el secador, pero no ha sido almacenado.
  -2:la papelera esta abierta y no se visualiza el secador, por lo que ha sido almacenado.

Todas las variables booleanas, inicialmente tienen el valor **false.** Las variables tipo **number**, inicialmente estarán en 0. Hay algunas variables que son del tipo **String**, dependiendo cual sea su función se inicializarán con un valor u otro

Estas son algunas de las variables. Como se ha comprobado la mayoría son del tipo Booleano, con dos estados. Algunas de estas variables tienen que compararse conjuntamente, ya que son dependientes, como por ejemplo: alien+nombre+disparado y escenariocambiado. Estas dos variables necesitan saber que valor tiene la otra para funcionar correctamente, ya que, los alien, solo pueden ser vistos si el escenario esta cambiado, pero puede ocurrir que el escenario este cambiado pero el alien no se visualice ya que se ha usado la pistola con el. Esto quiere decir, que es necesario que para que se vean los alien, la variable escenariocambiado este en true, pero no siempre que este en true, tienen porque verse. Por lo tanto en este caso se tendrán que evaluar los estados:

-**alien+nombre+disparado**=false && **escenariocambiado**=false
-**alien+nombre+disparado**=false && **escenariocambiado**=true
-**alien+nombre+disparado**=true && **escenariocambiado**=true
-**alien+nombre+disparado**=true && **escenariocambiado**=false

Estas variables, son leídas al principio, al llamar a la función **Objetos+Escenario()** en el escenario que les corresponde.

A continuación se explicarán las funciones más importantes. Todo el juego se basará en estas funciones, con modificaciones dependiendo de los objetos a los que se asocien.

7.2 FUNCIONES IMPORTANTES

La mayoría de estas pruebas están basadas en las aplicaciones que se hicieron para el capítulo 5.2, pero algunas de ellas, se hicieron por primera vez, durante el desarrollo de la aventura. Las más importantes son las siguientes. El resto de funciones del juego se basa en la estructura que poseen estas:

7.2.1 Interactividad de los vídeos

Se basa en la aplicación **5.2.3. Cambiar gestos de los vídeos.**

Ahora la interactividad del vídeo depende de los objetos. Al hacer click en un objeto, el vídeo evaluará la posición en la que se encuentra y dependiendo el valor, ejecutará una función u otra, que le trasladará a un cuepoint determinado para realizar los movimientos que indique.

La estructura ha cambiado ligeramente, pero la idea es la misma. Se ha creado una función que únicamente se encarga de que el vídeo, ejecute los movimientos de andar hacia la izquierda o hacia la derecha. Así a la hora de hacer click en un objeto no se tendrá que programar los cuepoints de andar, solo tendrá que llamarse a la función que se encarga de ello y solo habrá que encargarse de los movimientos propios para ese objeto.

Para que sea más realista la forma que tiene de interactuar el vídeo con los objetos, se han incluido cuepoints dentro de un mismo movimiento para que al pasar por ellos, los objetos cambien su estado en ese instante y no en el momento de pulsarlos .

Código:

A continuación se dará un ejemplo, para todo lo comentado de una forma más clara.

Se hace click en la *llave* y se evalúa la posición en la que se encuentra el personaje. En base al resultado se ejecuta una función u otra. Hasta aquí no ha cambiado nada.

```
llave.addEventListener(MouseEvent.CLICK,CogerLlave);

function CogerLlave(event:MouseEvent){

      PosicionesAulario2();

      if(posau2px==true){
            CogerlaLlave();
      }
      else if(posau2p1==true){
            IrApxyCogerLlave();
      }
}
```

La función **CogerLlave()** se ejecuta si se encuentra en la posición *px*. Esta es la función que hace el movimiento propio de coger y almacenar el objeto.

Dentro de ese movimiento se habrán introducido dos cuepoints. El primero llamado *"llave",* indica el momento en el que el protagonista estira la mano para coger el objeto. En ese instante , el objeto *llave* desaparece. El segundo llamado "almacenarllave" que corresponde con

el momento en el que el personaje hace el gesto de introducir el objeto en la mochila. En ese momento, se muestra la el objeto *llave* del inventario y se le da a su variable correspondiente, en este caso tengollave, el valor **true**. Los gestos asociados a los cuepoints pueden verse en las *figuras 7.5 y 7.6.*

```
function CogerlaLlave(){

      charliAulario2.seekToNavCuePoint("cogerllave");
      charliAulario2.addEventListener(MetadataEvent.CUE_POINT,
LlaveCogida);
}

            function LlaveCogida(eventObject:MetadataEvent):void {

                  if(eventObject.info.name == "llave") {
                        llave.visible=false;
                  }
                  if(eventObject.info.name == "almacenarllave") {
                        llavein.visible=true
                        tengollave=1;
                  }
                  if(eventObject.info.name == "cogerllavefin") {
                        QuedarseEnpx();
                  }
            }
```

Figura 7.5. Gesto cuepoint "llave"

Figura 7.6. Gesto cuepoint "almacenar llave"

Si en cambio se encuentra en la posición *p1*, ejecuta la función **IrApxyCogerLlave().**

Lo primero que hace es ejecutar la función que anteriormente se ha comentado, que se encarga de realizar únicamente los movimientos de andar. A esta función se le ha llamado **IrDep1Apx().**

Al ejecutarse esta función el vídeo salta al movimiento correspondiente de desplazarse a la izquierda y el objeto *sombra,* realiza un movimiento hacia el mismo lugar. Esta vez, en vez de colocar directamente la *sombra* en la marca de la posición final, se le ha dado un movimiento parejo al del vídeo.

```
function IrDep1Apx(){
//Se dirige al cuepoint con el movimiento de moverse a ala izquierda
      charliAulario2.seekToNavCuePoint("p1irapx");
//Se ejecuta el evento que hace desplazarse a la sombra
stage.addEventListener(Event.ENTER_FRAME,DesdeP1SombraPx);
}
      function LlegarApx(eventObject:MetadataEvent):void {
            /*Cuando detecta que ha llegado al cuepoint de final de
            movimiento se queda en esa posición*/
            if(eventObject.info.name == "p1irapxfin") {
                  PosicionesAulario2();
                  QuedarseEnpx();
            }
      }
      //Función que define el movimiento de la sombra
      function DesdeP1SombraPx(event:Event):void {

            sombraau2.x-=4;
            PosicionesAulario2();
            if (posau2px==true){
                  stage.removeEventListener(Event.ENTER_FRAME,DesdeP1So
            mbraPx);
            }
      }
```

Al haber hecho click en la llave, se detecta el cuepoint "llavep1irapxfin" que indica que ha terminado el movimiento de desplazamiento a la izquierda y por lo tanto se encuentra en *px*, por lo que llama a la función **CogerLlave().**

```
function IrApxyCogerLlave(){

      IrDep1Apx()
      charliAulario2.addEventListener(MetadataEvent.CUE_POINT,LlegarApx
      yCogerLlave);
      }
      function LlegarApxyCogerLlave(eventObject:MetadataEvent):void {
            if(eventObject.info.name == "llavep1irapxfin") {
                  PosicionesAulario2();
                  CogerlaLlave();
                  }
      }
```

Salvo por pequeñas modificaciones todos los vídeos tendrán la misma estructura, dependiendo de el objeto que intervenga. Además, de los movimientos de coger, existen otros como: disparar, hablar, etc. en los que se han introducido cuepoints para que los objetos se oculten o se muestren en el momento oportuno y no cuando se hace click.

7.2.2. Punteros del ratón

Esta función está basada en la aplicación **5.2.12 Cambiar el cursor.**

El funcionamiento es el mismo, al pasar por encima de un objeto determinado cambiará la imagen que se muestra del puntero del ratón. Para la aventura se dispondrá de cinco punteros diferentes:

Figura 7.7 Tipos de punteros

La diferencia que hay respecto a la aplicación realizada, es que en la aventura gráfica, nunca se verá el cursor por defecto, ya que se ha creado uno específico para el juego, que será visible en todo momento, excepto al pasar por las zonas determinadas. Por lo tanto se tendrá que ocultar el puntero por defecto desde el principio.

Otra novedad que hay que añadir es que se tendrá que dar la propiedad **mouseEnabled** a las instancias de los punteros. Esta propiedad hace que la instancia asociada no reciba ningún evento de ratón. Esto sirve para que únicamente se visualicen las imágenes en el momento que les corresponda, pero cuando surjan eventos de ratón no los reciban, de otro modo, surjirían problemas a la hora de visualizar las imágenes.

Código:

Primero se aplica a cada instancia del cursor la propiedad **mouseEnabled** y se ocultan todos los punteros salvo el que corresponde a *mccursor*, que será el cursor principal de la aventura.

```
mccursor.mouseEnabled=false;
mcmano.mouseEnabled=false;
mcboca.mouseEnabled=false;
mcpunto.mouseEnabled=false;
mcpies.mouseEnabled=false;

mcpunto.visible=false;
mcmano.visible=false;
mcboca.visible=false;
mcpies.visible=false;
```

Se añade al escenario un detector de eventos cada vez que se mueva el ratón, asociado a la función **Punteros()**. Esta función esconde el puntero por defecto y coloca los movieclips que se han creado en la posición del puntero de ratón.

Esta función debe colocarse en el primer fotograma de la aventura para que se detecte desde el principio.

```
stage.addEventListener(MouseEvent.MOUSE_MOVE,Punteros);

function Punteros(event:MouseEvent):void {

      Mouse.hide();
      mccursor.x=event.stageX;
      mccursor.y=event.stageY;
      mcmano.x=event.stageX;
      mcmano.y=event.stageY;
      mcboca.x=event.stageX;
      mcboca.y=event.stageY;
      mcpunto.x=event.stageX;
      mcpunto.y=event.stageY;
      mcpies.x=event.stageX;
      mcpies.y=event.stageY;
}
```

Por último, en cada escenario se llamará una función del tipo, **Punteros+NombreDelEscenario()**, que contendrá los detectores asociados al objeto que se quiere que cambie el cursor, para que se ejecute las funciones y se muestre el cursor correspondiente.

En este caso, al pasar por encima del objeto *llave*, se llamará a la función **mostrarmano()**, que oculta el cursor principal y muestra la mano y por consiguiente, al salir de encima de la llave se llamará a la función **nomostrarmano()** que restablecerá el cursor principal y ocultará la mano.

El resto de funciones que podrán llamarse para asociar los diferentes punteros son: **mostrarpies()**, **mostrarboca()** y **punterolapiz()**.

```
function PunterosAulario(){

      llave.addEventListener(MouseEvent.MOUSE_OVER,mostrarmano);
      llave.addEventListener(MouseEvent.MOUSE_OUT,nomostrarmano);
}
function mostrarmano(event:MouseEvent):void{

      mcmano.visible=true;
      mccursor.visible = false;
}
function nomostrarmano(event:MouseEvent):void{

      mcmano.visible=false;
      mccursor.visible = true;
}
```

A continuación se muestran cuatro imágenes con un cursor diferente dependiendo del objeto sobre el que estén.

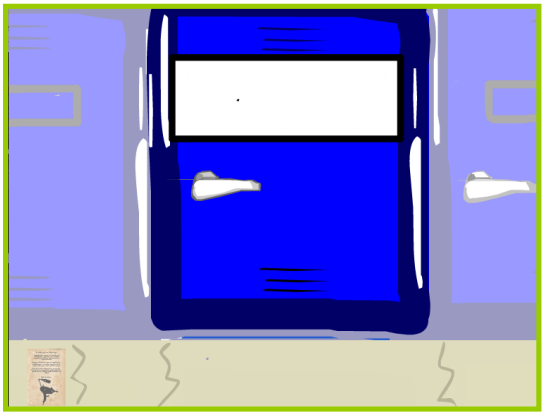


Figura 7.8. Detalle puntero de pizarra.

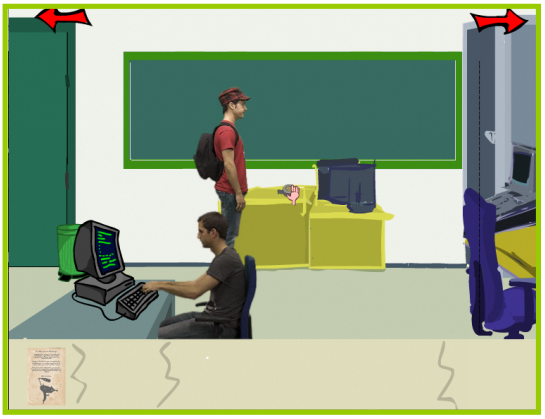


Figura7.9. Detalle puntero mano.

Figura 7.10. Detalle puntero pies.

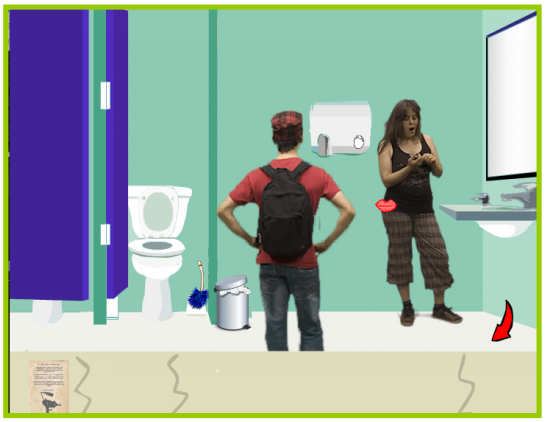


Figura7.11. Detalle puntero boca.

7.2.3 Texto del puntero

Se basa en la aplicación **5.2.10 Aparecer texto**.

Al colocar el ratón encima de determinados objetos, aparece el texto que se ha asociado a cada movieclip, a través de la propiedad text.

La programación que se ha usado, es exactamente la misma a la aplicación creada anteriormente.

Para que funcione correctamente, habrá que dar a cada objeto del escenario un texto con la propiedad text.

Ya que la función **mostrartexto()** se activa con el escenario cada vez que se mueve el ratón, habrá que tener especial cuidado, ya que si se incluye un objeto en el escenario y no se le aplica la propiedad **text**, al pasar por encima de un movieclip sin esta propiedad, a la salida aparecerá un error.

A los objetos que no necesitan que aparezca texto, se les aplicará " ". Por ejemplo *llave.text="".*

Esta función se utilizará para los objetos del inventario, y para las flechas del escenario. También dará información en determinadas partes del juego, como método de ayuda.

Figuras 7.11 y 7.12 Detalle texto puntero

Código:

Lo primero, se crea un formato de texto, con las propiedades de color, tamaño, alineación y tipo de letra. Se crea el campo de texto al que se le llamará *texto* y se añade al escenario y por último se adjudica el formato de texto creado a *texto*.

El texto inicialmente se ocultará.

```
//Formato del texto
var formatoTexto:TextFormat=new TextFormat();
formatoTexto.color=0x000000;
formatoTexto.size=14;
formatoTexto.align='center';
formatoTexto.font='Chalkduster';

//Se crea un campo de texto nuevo y se añade al escenario
var texto:TextField = new TextField();
addChild(texto);
//Se le da el formato creado, al campo de texto
texto.defaultTextFormat=formatoTexto;
texto.visible=false;
```

Se añade un detector de eventos del tipo **ENTERFRAME,** que se activará al ejecutar la película. Al detectarse el evento, el campo de texto se sitúa a 5 píxeles en x del puntero del ratón.

```
//Detector de evento de ratón, cuando se inicia la aplicación
stage.addEventListener(Event.ENTER_FRAME,movertexto);

function movertexto(event:Event):void{

      texto.x=mouseX+5;
      texto.y=mouseY;
}
```

Se asignará a cada objeto la propiedad text y se le dará un valor.

```
//Asignar a cada objeto la propiedad text
aulario.text="";
aulario2pasillo.text="Ir al pasillo";
```

Se añadirán al escenario dos detectores de evento, asociados a las funciones **mostrartexto()** y **ocultartexto()**. Cuando se pase por encima de un objeto se le asignará al campo de texto, el texto que lleve asociado el objeto y se mostrará. En el momento que no este encima, volverá a ocultarse.

```
/*Detector de eventos en el escenario, cuando se pase por encima de
un objeto y cuando se está fuera de el.*/
stage.addEventListener(MouseEvent.MOUSE_OVER,mostrartexto);
stage.addEventListener(MouseEvent.MOUSE_OUT,ocultartexto);}

/*Función que asigna al campo de texto creado, el texto asociado al
objeto y lo muestra*/
function mostrartexto(Event:MouseEvent):void{
      texto.text=Event.target.text;
      texto.visible=true;
}

/*Cuando no pase por encima de ningún objeto el campo de texto no
se verá*/
function ocultartexto(Event:MouseEvent):void{
      texto.visible=false
}
```

7.2.4 Escribir

Esta función esta asociada a la aplicación **5.2.13. Introducir texto.**

La dinámica es la misma, existirá un campo de texto en el que se podrá escribir. Habrá un objeto que al pulsarlo dependiendo de si se ha escrito la palabra correcta ocurrirá algo y sino no.

Esta parte del código está asociado a una taquilla, llamada *taquillaEscribir*, que se encuentra en el escenario. Al pulsar en esta taquilla, *figura 7.13*, se cumplen ciertos requisitos y se pasa al escenario donde hay una ampliación de esa taquilla *figura 7.14*. En este escenario se encuentra el campo de texto y el objeto que permite hacer la comprobación

Figura 7.13. Captura escenario con taquillas

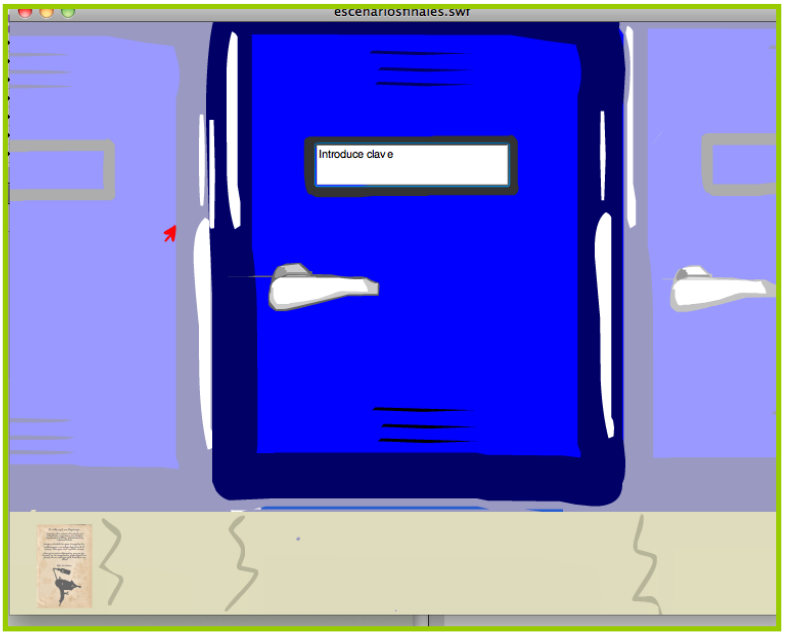


Figura 7.14. Detalle taquilla escribir

A continuación se explicarán los tres pasos: ir al escenario con la taquilla ampliada, introducir el texto, y comprobar el texto.

Código:

Al hacer click en la taquilla, se accede al escenario donde se podrá introducir el texto. El cuadro de texto donde se escribirá se llamará *clave*.

```
aulario.taquillaEscribir.addEventListener(MouseEvent.CLICK,zoomTaquil
laEscribir);

function zoomTaquillaEscribir (event:MouseEvent):void{
      /*Se pone Charli en stop para que no ocurran errores al pasar
      al otro escenario*/
      charliAulario2.stop();
      texto.visible=false;
      taquilla1.visible=false;
      gotoAndStop(38);
      clave.visible=true;
}
```

Una vez en este escenario, habrá que introducir cierta palabra para abrir la taquilla. Para comprobar si la palabra es la correcta o no, se debe pulsar en el objeto *manillaes.*

La manilla esta asociada a una función llamada **comprobarClave()**, que se ejecuta al hacer click en ella.

Esta función, se encarga de comparar si el texto del campo de texto *clave* es la palabra correcta, en este caso, la palabra que hay que introducir es “limon”.

En el caso de que la palabra introducida haya sido limón, se reproducirá un sonido de cerradura y se pondrá en **true** la variable taquillaEscribirAbierta, correspondiente al estado de la taquilla. Sino es correcta, la variable permanecerá en **false.** A continuación, se ejecutará la función **comprobarTaquillaEscribir().**

```
manillaes.addEventListener(MouseEvent.CLICK, comprobarClave);

function comprobarClave (event:MouseEvent):void{

      comprobarTaquillaLLave()

      if(clave.text=="limon"){
            sonidoCerradura.play();
            taquillaEscribirAbierta=true;
            comprobarTaquillaEscribir()

      }
      else{
            comprobarTaquillaEscribir()
      }
}
```

Con la función **comprobarTaquillaEscribir()**, se accede al escenario inicial (donde se encuentran las taquillas). Esta función evalúa conjuntamente, las variable asociadas al objeto *limón* y a la *taquilla.*

Hay tres posibles estados:

- Que la taquilla haya sido abierta pero el limón no se haya cogido.
- Que la taquilla haya sido abierta y el limón haya sido cogido.
- Que la taquilla no haya sido abierta y por lo tanto el limón no se haya cogido.

```
function comprobarTaquillaEscribir(){
      gotoAndStop(4);
            //Si la taquilla esta abierta y no se ha cogido el limon
            if((taquillaEscribirAbierta==true)&&(tengolimon==false)){
                  //El objeto no se ve
                  aulario.taquillaEscribir.visible=false;
                  manillaes.visible=false;
                  limon.visible=true;
                  limonin.visible=false;
            }
            else if
      ((taquillaEscribirAbierta==true)&&(tengolimon==true)){

                  aulario.taquillaEscribir.visible=false;
                  manillaes.visible=false;
                  limon.visible=false;
                  limonin.visible=true;

            }
            else if
      ((taquillaEscribirAbierta==false)&&(tengolimon==false)){

                  aulario.taquillaEscribir.visible=true;
                  manillaes.visible=true;
                  limon.visible=false;
                  limonin.visible=false;
            }
}
```

7.2.5. Dibujar

Esta asociada a la aplicación 5.2.15. Dibujar.

Esta taquilla, sigue la misma filosofía que la anterior, solo que en vez de escribir, en esta habrá que realizar un dibujo para abrirla.

Al hacer click en la taquilla, denominada *taquillaDibujar*, se accede a otro escenario que contendrá un espacio donde se podrá dibujar al que se le llamará *pizarra* y el objeto *manilladibu*, que permitirá hacer la comprobación del dibujo *figura 7.15*.

Para abrir la taquilla se deberá hacer un dibujo, que podrá verse al hacer click en el objeto *lentillas figura 7.16* (más tarde se explicará la función que tiene). Cuando el dibujo se realice correctamente, al pulsar en la manilla se volverá al primer escenario y se visualizará la taquilla abierta y el objeto que esconde *figura7.17* y en el caso contrario la taquilla se visualizará como cerrada.

Para detectar si el dibujo es correcto o no, se han colocado 7 puntos de colisión, con el alpha 0, es decir transparentes. En el momento que el objeto *lápiz*, haya colisionado con todos, es cuando el dibujo se dará por bueno.

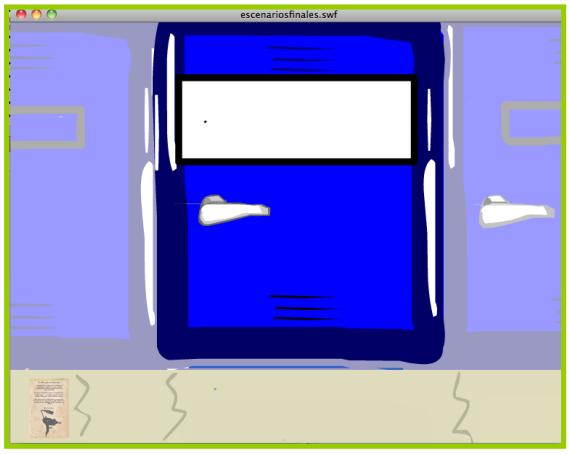


Figura 7.15. Detalle taquilla dibujar

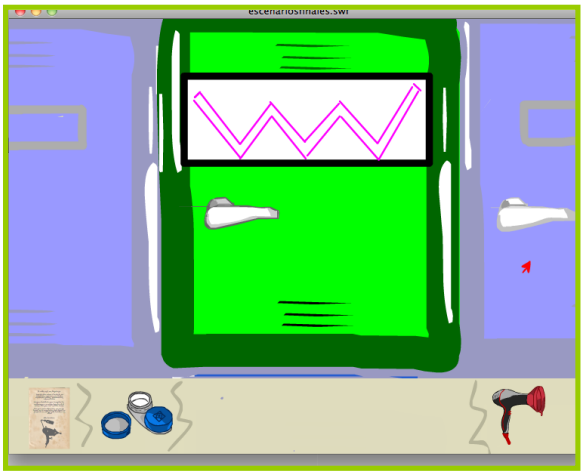


Figura 7.16 Detalle taquilla dibujar con escenario cambiado

Figura 7.17. Escenario con taquilla dibujar abierta

El primer paso y el último, es decir acceder al escenario con la taquilla ampliada y comprobar la taquilla, son muy similares a la estructura de la *taquillaEscribir*, solo cambian los objetos y variables de los que depende. Por lo que, a continuación, se explicará únicamente la parte de dibujar y detectar los puntos de colisión.

Código

Se crea un nuevo objeto que será de la clase Shape, al que se le llamará *lapiz.*

Se crea la función dibujar. En ella, se define como va ser el tipo de línea, mediante el método **lineStyle()**. Los parámetros que se introducirán serán el color y el grosor, en este caso, negro y 8.

A continuación, se añaden los eventos de ratón correspondientes al objeto *pizarra*.

El objeto *guía,* es el movieclip creado para saber que dibujo hay que realizar. Se visualizará al pulsar en el objeto *lentillas*. A este objeto también se le añadirán eventos de ratón para que cuando esté visible, pueda dibujarse sobre el. Se hará lo mismo con los puntos de colisión llamados *p1, p2, p3*...hasta *p7*. (Aquí solo se muestra el punto *p1*)

```
var lapiz:Shape= new Shape();

function Dibujar(){
//Se define como va a ser el tipo de líneas en "lapiz".
//En este caso seria grosor 8, y color negro
      lapiz.graphics.lineStyle(8,0x000000);

      //Se añaden los eventos a la pizarra que será donde se pueda
dibujar
      pizarra.doubleClickEnabled = true;
      pizarra.addEventListener(MouseEvent.MOUSE_OVER,punteroLapiz);
      pizarra.addEventListener(MouseEvent.CLICK,punteroLapiz);
      pizarra.addEventListener(MouseEvent.MOUSE_DOWN,siPintar);
      pizarra.addEventListener(MouseEvent.MOUSE_UP,noPintar);
      pizarra.addEventListener(MouseEvent.DOUBLE_CLICK,borrar);
      pizarra.addEventListener(MouseEvent.MOUSE_OUT,noPintar);

      guia.addEventListener(MouseEvent.MOUSE_OVER,punteroLapiz);
      guia.addEventListener(MouseEvent.MOUSE_DOWN,siPintar);
      guia.addEventListener(MouseEvent.CLICK,punteroLapiz);

      p1.addEventListener(MouseEvent.MOUSE_DOWN,siPintar);
      p1.addEventListener(MouseEvent.MOUSE_OVER,punteroLapiz);
      p1.addEventListener(MouseEvent.CLICK,punteroLapiz);
}
```

Al moverse el puntero por encima de la pizarra o hacerse click sobre ella, se activará la funcion **punterolapiz()**, esta función deja ver el puntero *mcpunto*, que hará de lápiz para la pizarra.

```
function punteroLapiz(e:Event):void{

      mcpunto.visible=true
      mccursor.visible=false;
}
```

La función **nopintar()** se ejecuta cuando no se pulsa el ratón. Lo que ocurre es que elimina el detector de eventos asociado a la función **pintar**, que se explicará más adelante.

```
function noPintar(e:Event):void{
//Al soltar el ratón, elimino el evento para pintar lineas
      removeEventListener(Event.ENTER_FRAME,pintar);
      mcpunto.visible=false
      mccursor.visible=true;
}
```

La funcion **sipintar()** es llamada cuando se hace click sobre la pizarra. Esta función lo primero que hace es colocar el punto de dibujo (donde se comenzará a trazar la línea) en el puntero del ratón, mediante el método **moveTo()** y se visualizará el puntero *mcpunto*. Esta función a su vez, llamará a un evento **ENTERFRAME**, que estara asociado a la funcion **pintar**.

```
function siPintar(e:Event):void{
/*Cuando el usuario presiona el botón izquierdo, el puntero debe
empezar a pintar. El punto de inicio de ese nuevo trazo, debe ser
el del puntero del ratón, por lo que muevo la posición del dibujo
con moveTo*/
      lapiz.graphics.moveTo(mouseX,mouseY);
      mcpunto.visible=true
      mccursor.visible=false;
//Añado un listener para que se pinten lineas siguiendo al puntero
      addEventListener(Event.ENTER_FRAME,pintar);
}
```

En la función **pintar()**, se le asigna al objeto *lápiz* el método **lineTo()**. Este método traza un línea recta desde el punto de dibujo actual (el puntero del ratón) hasta las coordenadas especificadas en los dos parámetros de la llamada al método, en este caso son las coordenadas del ratón. Por lo tanto, se irá dibujando una línea por donde pase el cursor, mientras se mantenga pulsado.

Mientras se ejecuta esta función se comprueba si lo puntos han colisionado con *lápiz.* Esto se hace mediante el método **complexHitTestObjet()** que realiza una colisión a nivel de píxeles[F44]. Es necesario usar este método y no **HitTestObject()** ya que este método realiza la colisión mediante el área de su contenedor *figura 7.18.,* por lo que la comprobación no se haría correctamente

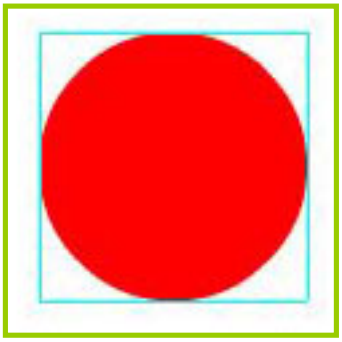


Figura 7.18. Rectángulo contenedor del objeto

En el momento que *lápiz* colisione con todos los puntos de contacto, se oirá un sonido de cerradura y se cambiara el valor de la variable taquillaDibujarAbierta a **true**.

```
function pintar(e:Event):void{
//La función lineTo pinta una línea en el punto (x,y)
      lapiz.graphics.lineTo(mouseX,mouseY);
      mcpunto.visible=true
      mccursor.visible=false;
      //Detección de los puntos de colisión
      if((HitTest.complexHitTestObject(lapiz,p1))&&(HitTest.complexH
itTestObject(lapiz,p2))&&(HitTest.complexHitTestObject(lapiz,p3))&&
(HitTest.complexHitTestObject(lapiz,p4))&&(HitTest.complexHitTestOb
ject(lapiz,p5))&&(HitTest.complexHitTestObject(lapiz,p6))&&(HitTest
.complexHitTestObject(lapiz,p7))){

            taquillaDibujarAbierta=true;
            sonidoCerradura.play();
      }
}
```

Por ultimo la función **borrar**, se ejecuta al hacer doble click sobre la pizarra. Esta función elimina lo que se ha dibujado, y restablece el tipo de línea con su grosor y color.

```
function borrar(e:Event):void{
      mcpunto.visible=true
      mccursor.visible=false;

/*La función clear borra lo que se haya dibujado en el objeto
graphics, y restablece la configuración de grosor y color*/
      lapiz.graphics.clear();
      lapiz.graphics.lineStyle(8,0x000000);
}
```

Para volver al escenario, se debe pulsar en el objeto *manilladibu*. Este objeto tendrá asociado una función similar a la explicada en la *taquillaEscribir*, que comprobará las variables de la taquilla y del objeto que estén relacionado con la *taquilladibujar*, para dejarlos en el estado que les corresponde.

7.2.6 Arrastrar objetos

Esta función esta basada en la aplicación **5.2.7. Coger objetos, arrastrar y colocar**.

Esta función permite arrastrar objetos con el ratón para poder combinarlos o usarlos con otros objetos, de modo que al soltar el ratón, evalúa si ha colisionado con el objeto correcto o no lo ha hecho. Dependiendo si colisiona o no, se realizarán unas acciones u otras.

La función mantiene la misma estructura que la creada. Las diferencias dependen de los objetos que intervengan en ella. Por ejemplo esta función se usa en **UsarLLave()**, **Combinargato()**, **Usargato()** o **ArrastrarPistolaAgua()**, entre otras.

Código:

Se crea una función que engloba a las funciones **Arrastrar+Objeto()** y **Noarrastrar+Objeto()**. De esta forma, solo se tendrá que llamar a una función.

```
function CombinarSecadorConEmbudo(){

      embudoin.addEventListener(MouseEvent.MOUSE_DOWN,ArrastrarEmbud)
      embudoin.addEventListener(MouseEvent.MOUSE_UP,NoArrastrarEmbud)
```

La función **Arrastrar+Objeto()** simplemente permite arrastrar el objeto al pulsarlo.

```
/*Esta función se ejecuta cuando se aprieta el botón izquierdo del
ratón
      function ArrastrarEmbudo(Event:MouseEvent):void{
            embudoin.startDrag();
      }
```

La función más importante es **Noarrastrar+Objeto(),** ya que es donde se evalúa si el objeto a colisionado con el correspondiente, en el momento en el que se suelta el ratón.

En este ejemplo se evalúa si el *embudo* ha colisionado con el *secador figura 7.19*, en el caso de que sí, se ocultan el secador y el embudo y se muestra el objeto que será combinación de los dos *figura 7.20*. Además, la variable tengosecabudo, cambia de estado, para que al cambiar de escenario, se lea y mantenga los estados de los objetos. En el caso de que no hayan colisionado, el *embudo* vuelve a su posición correspondiente y se deja de arrastrar.

```
/*Esta función se ejecuta cuando se suelta el botón del ratón
      function NoArrastrarEmbudo(Event:MouseEvent):void{    /*Si se
cumplen las dos condiciones entonces se ejecutarán las líneas
siguientes*/
            if(embudoin.hitTestObject(secadorin)){
                  secabudo.x=675;
                  secabudo.y=525;
                  secadorin.visible=false;
                  embudoin.visible=false;
                  secabudo.visible=true;
                  tengosecabudo="tengo secabudo";
            }//Si no se cumplen las condiciones:
            else{
/*Se dejará de arrastrar y la llave se coloca en la posición
indicada*/
                  embudoin.x =545;
                  embudoin.y = 500;
                  embudoin.stopDrag();
            }
      }
}
```

Figura 7.19. Detalle arrastrar embudo sobre secador

Figura 7.20. Detalle nuevo estado combinación embudo y secador

7.2.7 Uso de filtros

Esta función es completamente nueva, se creó durante el desarrollo de la aventura. Está basada en la idea de la aplicación **5.2.9. Cambiar el escenario,** pero se cambió completamente debido a que, el hecho de crear dos imágenes para cada escenario resultaba muy incómodo. Por eso, se eligió la forma de cambiar los colores de la propia imagen. Esto se realizó aplicando un filtro a la imagen.

Así únicamente, se necesitará una imagen como fondo de escenario, a la que en ciertos momentos se le aplicará un filtro y cambiará sus colores.

La mecánica es la siguiente. Sobre la imagen normal figura *7.21*. al hacer click sobre el objeto llamado *lentillas,* se aplica un filtro (más delante se explicará como funciona) a la imagen de fondo del escenario *figura 7.21*. También habrá objetos que se oculten y otros nuevos que se muestren. Para volver al estado inicial se pulsará el objeto *quitarlentillas* que eliminará el filtro aplicado.

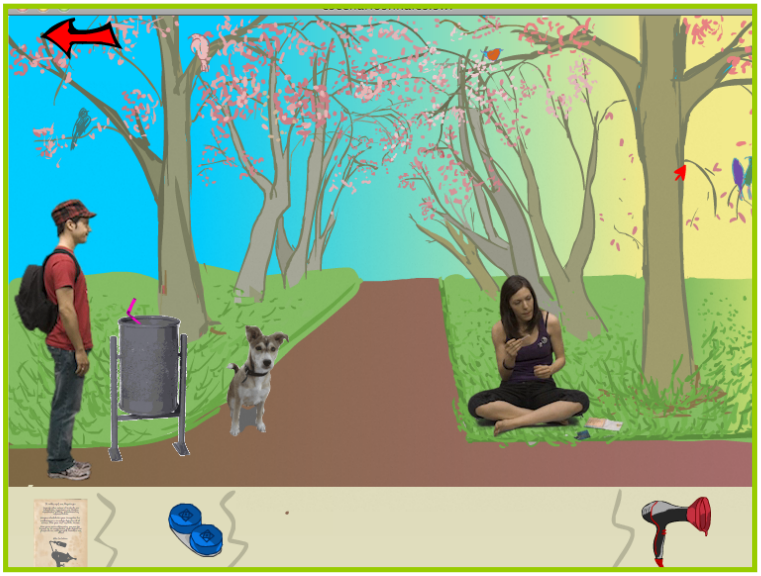


Figura 7.21. Escenario normal

Figura 7.22. Escenario con filtro

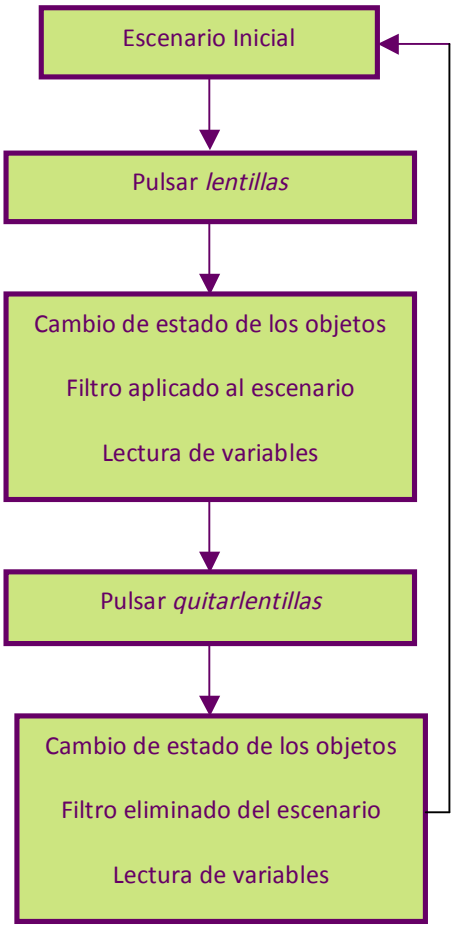


Diagrama 7.1. Uso de filtros

A continuación se explica el proceso paso a paso:

Código:

Al hacer click en las *lentillas figura 7.23.* se llamara a la función **cambiar+Escenario().**

```
lentillasinext.addEventListener(MouseEvent.CLICK, cambiarExterior);
quitarext.addEventListener(MouseEvent.CLICK, QuitarExterior);
```

La función **cambiar+Escenario()**, realiza las acciones oportunas, como ocultar el objeto *lentillas*, mostrar el objeto *quitarlentillas* y poner la variable escenariocambiado en **true**. Esto último permitirá que si el jugador cambia de escenario, se siga viendo con el filtro aplicado.

Figura 7.23. Detalle objeto lentillas

También se evalúan las variables asociadas a los objetos que están relacionados con el cambio de escenario, ya que en el caso que se haya realizado alguna acción con o sin el filtro aplicado, se deben mantener los estados correspondientes.

Por último, llamará a la función **cambiar+Escenario().**

```
function cambiarExterior(event:MouseEvent) {

      cambiarext();
      lentillasinext.visible=false;
      quitarext.visible=true;
      escenariocambiado=true;
      //Lectura de variables asociadas a los objetos y al escenario
      if((alienbrutadisparada==true)&&(escenariocambiado==true)){
            alienBruta.visible=false;
            Bruta.visible=false;
            Carol_mc.visible=false;
      }
      else
if((alienbrutadisparada==true)&&(escenariocambiado==false)){
            alienBruta.visible=false;
            Bruta.visible=false;
            Carol_mc.visible=false;
      }else
if((alienbrutadisparada==false)&&(escenariocambiado==true)){
            alienBruta.visible=true;
            Bruta.visible=false;
            Carol_mc.visible=false;
      }else
if((alienbrutadisparada==false)&&(escenariocambiado==false)){
            alienBruta.visible=false;
            Bruta.visible=true;
      }
```

La función **cambiar+Escenario()** aplica el filtro al escenario. Para aplicar el filtro hace falta crear una matriz 4x5. Este filtro de la matriz de colores separa cada píxel de origen en sus componentes rojo, verde, azul y alfa. Para calcular el resultado de cada uno de los cuatro canales, el valor de cada píxel de la imagen se multiplica por los valores de la matriz de transformación. Opcionalmente, se puede añadir un desplazamiento que oscile entre -255 y 255 para cada resultado (el quinto elemento de cada fila de la matriz). El filtro combina cada componente de color en un único píxel y escribe el resultado [F45-F46].

Para aplicar el filtro al escenario, se utiliza la propiedad filters. Esta propiedad no modifica el objeto y se puede eliminar el filtro borrando la propiedad filters.

Se crea una instancia ColorMatrixFilter nueva con los parámetros especificados de la matriz.

```
function cambiarext(){

      var matrix3:Array = new Array ( );

      matrix3 = matrix3.concat ( [ 0 , 1 , 0 , 0 , 0 ] ) ; // red
      matrix3 = matrix3.concat ( [ 0 , 0 , 1 , 0 , 0 ] ) ; // green
      matrix3 = matrix3.concat ( [ 1 , 0 , 0 , 0 , 0 ] ) ; // blue
      matrix3 = matrix3.concat ( [ 0 , 0 , 0 , 1 , 0 ] ) ; // alpha

var
my_filterexterior:ColorMatrixFilter=newColorMatrixFilter(matrix3);
      exterior.filters = [ my_filterexterior ];
```

Al hacer click en el objeto *quitarlentillas figura 7.24,* se llama a la función **Quitar+Escenario().**

Figura 7.24. Detalle objeto quitar lentillas

Esta función elimina el filtro y restablece los valores que había antes de aplicar el filtro. La variable escenariocambiado vuelve a ser **false** y a continuación, se analizan los estados de cada objeto para dejarlos en el que les corresponde.

```
function QuitarExterior(event:MouseEvent) {

      lentillasinext.visible=true;
      quitarext.visible=false;
      escenariocambiado=false;
      exterior.filters = [];
      //Lectura de variables asociadas a los objetos y al escenario
      if((alienbrutadisparada==true)&&(escenariocambiado==true)){
            alienBruta.visible=false;
            Bruta.visible=false;
            Carol_mc.visible=false;
      }
      else
if((alienbrutadisparada==true)&&(escenariocambiado==false)){
            alienBruta.visible=false;
            Bruta.visible=false;
            Carol_mc.visible=false;
      }
      else
if((alienbrutadisparada==false)&&(escenariocambiado==true)){
            alienBruta.visible=true;
            Bruta.visible=false;
            Carol_mc.visible=false;
      }
      else
if((alienbrutadisparada==false)&&(escenariocambiado==false)){
            alienBruta.visible=false;
            Bruta.visible=true;
```

7.2.8. Cargar SWF externo.

Esta función ha sido creada por primera vez durante el desarrollo de la aventura.

Esta función se encarga de cargar un SWF en un determinado momento, después de realizar una acción.

Se utilizará dos veces, debido a que la aventura está formada por tres SWFs. Se llamará al principio, después de la introducción para cargar el juego y al final, cuando se termine el juego para cargar la animación final.

Para ello, se deben tener los SWF s en la misma carpeta que la aplicación Flash.

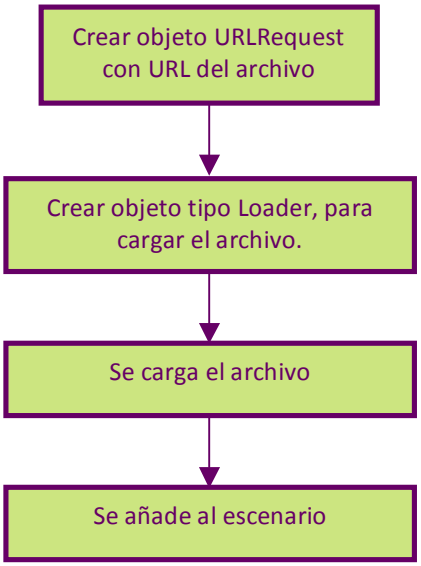


Diagrama 7.2. Cargar SWF externo

Código:

Se crea la función **Cargar()**. En ella se crea un nuevo objeto URLRequest, llamado *request*, con la URL del archivo externo. El objeto URLRequest pemite hacer la solicitud HTTP al servidor para transferir los archivos a descargar.

```
function Cargar(){

var request:URLRequest = new URLRequest("Final.swf");
```

A continuación, se crea un objeto Loader, llamado *cargar*. Estos objetos permiten cargar archivos dentro de una animación flash.

Mediante el método **load()** se carga el archivo en sí y con el método **addChild()** se agrega al escenario para poder ser visualizado

```
//Se crea un objeto del tipo Loador, que sirve para cargar archivos
externos

var cargar:Loader = new Loader()
//A ese objetos se le da la propiedad cargar y se le asigna el objeto
request que es el swf que se quiere cargar
cargar.load(request);
//se añade el objeto al escenario para ser visualizado
addChild(cargar);
}
```

7.2.9 Incluir sonido

Se basa en la aplicación **5.2.14 Reproducir sonidos.**

Para esta función se ha añadido la novedad de reproducir un sonido infinitas veces, esto se usará para el sonido de fondo de la aventura.

Para poder incluir un sonido en Flash, a través de ActionScript 3.0. Primero se importa el sonido a la biblioteca de Flash y se accede a las propiedades *figura 7.25.* Ahora habrá que crear una clase propia para el sonido. En el panel de propiedades, en la parte de **Linkage,** debe seleccionarse **Export for ActionScript** y en el cuadro de **Class**, escribir el nombre de clase que se quiere dar al sonido, en este caso se le llamará Fondo *figura 7.26.*

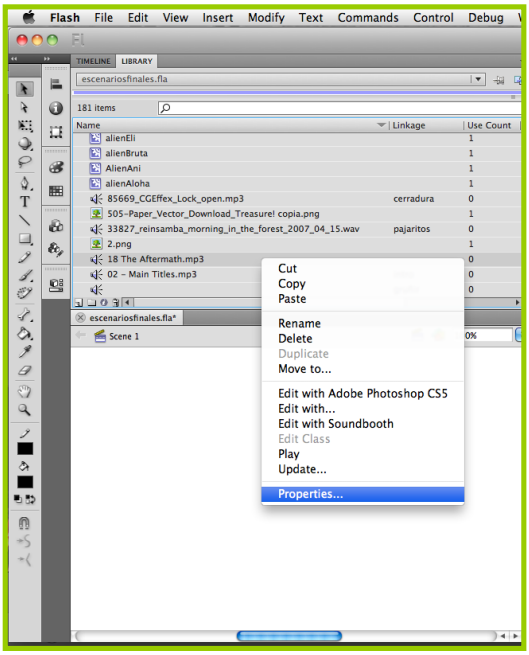


Figura 7.25 Importar sonido a biblioteca

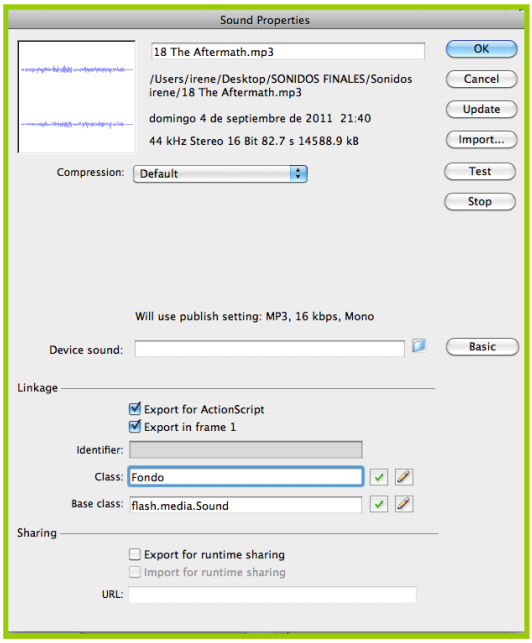


Figura 7.26. Propiedades del sonido

Código:

Para reproducir el sonido durante la ejecución de la aventura gráfica se seguirán los siguientes pasos..

Primero se crea una variable de la clase del sonido que se ha creado antes. Esto se realiza de la siguiente manera:

```
//Se ha creado previamente la clase de el sonido que se quiere
reproducir
var musicadeFondo:Fondo=new Fondo();
```

Se crea un canal de sonido llamado *canal_fondo*, que permite controlar el archivo de sonido que se reproduzca.

En el método **play()** se introducen dos parámetros. El primero indica la posición en la que se quiere que comience el sonido y el segundo indica el número de repeticiones que se quiere, se pondrá el máximo valor.

```
//Se crea un canal de sonido
var canal_fondo:SoundChannel;
/*El método play recibe 2 parámetros, la posición donde inicia el
sonido y el número de loops*/
canal_fondo=musicadeFondo.play(0,int.MAX_VALUE);
```

Capítulo 8: CONCLUSIONES Y LÍNEAS FUTURAS

Las conclusiones que se han obtenido tras la realización del proyecto son muy variadas, por eso se ha hecho una diferenciación entre conclusiones generales, técnicas y personales:

8.1 GENERALES

En primer lugar destacar la importancia de la búsqueda de información previa al comienzo del desarrollo del proyecto. En la primera fase se realizó una recopilación de toda la información que se consideró necesaria para el desarrollo del proyecto. Esto englobaba todos los aspectos relacionados con las aventuras gráficas, los programas que iban a ser utilizados y las posibilidades que ofrecían cada uno, sobre la técnicas de grabación que se iba a realizar así como tutoriales y manuales acerca de las tecnologías utilizadas.

En segundo lugar y en relación con lo anterior, se estructuró todo el trabajo en base a la información recopilada. Fue un proceso largo, porque toda esa información había que ordenarla y clasificarla, pero este esfuerzo mereció la pena, ya que aclaró muchas ideas sobre como se podía estructurar el proyecto, lo que era posible hacer o no y lo más importante, como hacerlo de la mejor manera posible.

Dentro de la estructura del proyecto, la parte más importante, fue el planteamiento y realización de las pruebas previas. Fue un trabajo largo y laborioso pero era vital ya que dio sus frutos a la hora de desarrollar la aventura debido a que muchas de las aplicaciones usadas no costó mucho implementarlas en el juego porque ya estaban creadas y la grabación del personaje final se realizó sin complicaciones ya que se tenían todos los conceptos de grabación, analizados y asimilados.

Otra lección aprendida y que ha sido de mucha ayuda es el no tener prisa. Es decir, no empezar con algo, hasta no tener terminado lo anterior. Esto ha servido para llevar el proyecto durante todo el tiempo de una forma ordenada. Cuando se diseñaron las pruebas, primero había que plantearlas, después realizarlas y por ultimo analizarlas. Hasta que estos tres pasos no se hubieran cumplido no se pasaba a la siguiente prueba. De igual forma, hasta no se realizaron todas las pruebas no se comenzó a diseñar la aventura gráfica final.

Esto último permitió ir realizando los documentos necesarios que posteriormente se incluirían en este proyecto. De esta forma se agilizó mucho el trabajo final de redacción de la memoria.

Por último decir que la realización de una aventura gráfica de este tipo es un proceso largo y laborioso y más si nunca se ha hecho. Se puede equiparar a la realización de una pequeña película. Con la diferencia que este proyecto puede dividirse en dos partes, que serían las pruebas realizadas, con la fase de preproducción, que sería su planteamiento inicial, la producción que sería el desarrollo propio de las pruebas y el análisis de cada una que correspondería con la fase de postproducción. A su vez, todo el proceso desarrollo de la aventura gráfica también puede dividirse en estas tres partes. Primero se realizó un guión técnico y otro artístico que corresponde con la fase de preproducción, más tarde se procedió a la realización de toda la programación y por ultimo, se realizaron todos los escenarios y objetos necesarios para que la aventura tuviera una interfaz más atractiva.

8.2. TÉCNICAS

Lo primero que cabe destacar, es la importancia del estudio previo del lenguaje de programación ActionScript 3.0. y el proceso de realización de las aplicaciones. Esto ha permitido trabajar con gran fluidez, dentro de los conocimientos adquiridos, durante el desarrollo de la y sobre todo ha otorgado la capacidad de resolver los problemas que iban surgiendo de un modo más rápido y sencillo.

También ha sido importante diseñar y plantear una estructura de programación clara y ordenada, así como la realización previa a la programación de diagramas de bloques, que hacen mas clara la visón de lo que se quiere programar.

La forma en la que se ha desarrollado la aventura ha sido mediante la realización de funciones dentro del entorno de Flash. Otra posible forma de realizar la programación es mediante la creación de clases en archivos externos. Puede que de esta forma, la aventura hubiera estado estructurada de una forma más profesional, pero al ser posible la opción de realizarla mediante funciones, y sobre todo después de todo el tiempo que duró el proceso de realización de pruebas, se descartó la idea. Además, el resultado final de la aventura ha sido el que se esperaba ya que ha podido realizarse todo lo que se había pensado.

La ultima conclusión es sobre la aventura en si. Se diseño con siete escenarios, que en un principio parecía un número asequible. Conforme se iba desarrollando la aventura, fue más visible el hecho de que siete escenarios resultaron ser mucho, ya que el número de personajes y de objetos que programar también aumentaba. La dificultad de realizar la aventura con más o menos escenarios es la misma, pero el desarrollo es más lento conforme más escenarios y más objetos se tienen. En su favor hay que decir, que al haber más escenarios, más objetos y más personajes, la aventura gráfica resulta más atractiva ya que hay más posibilidades de juego.

8.3 PERSONALES

Todo el proyecto se ha realizado con actitud muy positiva. Ya que el tema en si de realizar una aventura gráfica resultaba muy atractivo. Además, el hecho de incluir vídeo real, ofrecía la oportunidad de aprender conceptos, que engloban el conocimiento del uso de una cámara semiprofesional, la grabación en un estudio y su posterior edición, que es algo que interesaba mucho aprender. A pesar de las complicaciones que pudieron surgir y del largo proceso de desarrollo y ejecución de todas las ideas del proyecto, la estructura ideada y la forma ordenada en la que se ha llevado a cabo, ha facilitado todo este proceso y ha hecho que el trabajo se haya realizado con mucha fluidez y por lo tanto con mucho optimismo.

Por último destacar todo lo que se ha aprendido durante estos 12 meses, con este proyecto. Lo más importante es haber conseguido comprender un lenguaje de programación nuevo, ya que era la parte más temida. Conseguir traducir ideas que en un principio solo se encuentran en un papel, al lenguaje de ordenador, era uno de los principales objetivos personales en el desarrollo del proyecto. Como se ha dicho antes, también se ha aprendido a manejar una cámara semiprofesional, a modificar los parámetros necesarios dependiendo lo que se quiere grabar y a realizar una buena iluminación para la grabación en un estudio. Personalmente este tema resultaba muy interesante y en vista de los resultados se ha conseguido el objetivo.

BIBLIOGRAFÍA

INFORMACIÓN SOBRE AVENTURAS GRAFICAS

-Explicación del sistema SCUMM usado por la compañía LucasArt.

[A01]
http://es.wikipedia.org/wiki/SCUMM

-Explicación del sistema point&click.

[A02]
http://es.wikipedia.org/wiki/Point-and-click

-Página sobre la historia de las aventuras gráficas, un poco escueta, pero viene bien para hacerte una idea de cómo empezó todo.

[A03]
http://finalbossstudio.blogspot.com/2010/06/historia-de-las-aventuras-graficas-i.html

-Guía en inglés sobre los pasos que hay quedar para crear un aventura gráfica.

[A04]
http://www.adventureclassicgaming.com/index.php/site/features/105

-Página donde te marca los pasos que hay que dar para crear una aventura gráfica y el material que debes usar. Interesante para saber un poco por donde hay que empezar.

[A05]
http://www.cristalab.com/tutoriales/manual-de-animacion-en-flash-c144l/#personajes

-Artículo sobre las aventuras gráficas y sobre todo sobre los puzzles, los tipos que existen.

[A06]
http://la-aventura.net/fan-articles/aplicacion-de-la-teoria-del-puzzle

-Documento que explica el como crear un guión para un videojuego.

[A07]
http://www.slideshare.net/valtovar/diseo-de-video-juegos-guin

-Página donde te muestra los distintos motores y programas que existen para crear aventuras gráficas.

[A8]
http://www.sonidosimaginarios.es/manuelgertrudix/?p=208

Aventuras gráficas en Flash

-La ínsula de sancho.

[A9]
http://www.vidaextra.com/aventura-plataformas/la-insula-de-sancho-gran-aventura-grafica-en-flash

-The crows.

[A10]
http://www.thecrows.es/

-The Terrific Menace of the Invaders from Audiogalaxy.

[A11]
http://www.genereavventura.com/eng/play.html

-Aquaria.

[A12]
http://www.so-room.com/contents/flashgames/aquaria2.php

-Gateway.

[A13]
http://www.freeonlinegames.com/game/gateway.html

-The goat in the grey fedora.

[A14]
http://www.otterarchives.com/bounty2/bounty2.html

-Escapar de la habitación

[A15]
http://mofuya.com/flash/swan_en.htm

-Páginas con muchos juegos en flash

[A16]
http://freegames.1up.com/adventure-games/

[A17]
http://www.mofunzone.com/online_adventure_games.shtml

[A18]
http://www.newgrounds.com/game/adventuregames

-Dos blogs sobre la creación de una aventura gráfica, los pasos que estas personas han ido dando y la evolución del juego. El segundo enlace es mejor.

[A19]
http://diariobizarre.blogspot.com/2008_03_01_archive.html

[A20]
http://newbitsgames.blogspot.com/2010_04_01_archive.html

Flash con imágenes reales

-Página que recopila varios vídeos interactivos con imágenes y video real. Muy interesante.

[A21]
http://www.vincentmorisset.com/

-Anuncio muy bueno de la marca de ropa STADIUM, hecho con imágenes reales y vídeo.

[A22]
http://demo.fb.se/e/stadium/running/

-Anuncio de Ikea hecho en flash, imágenes que giran.

[A23]
http://demo.fb.se/e/ikea/dreamkitchen2/site/default.htm

-Anuncio de volvo.

[A24]
http://demo.fb.se/e/volvov50/site/start.html

-Vídeo interactivo muy bueno en flash, en el que puedes interactuar con las manos y la cabeza del hombre.

[A25]
http://www.taringa.net/posts/animaciones/2017581/Video-interactivo-en-flash-The-Arcade-Fire-Neon-Bible.html

-Página donde vienen muchos videos interactivos en flash, algunos de ellos con imágenes reales.

[A26]
http://www.albinoblacksheep.com/flash/interactive/

VÍDEO EN FLASH

Componente flvplayback.

[V1]
http://www.codigometropoli.com/componente-flvplayback/

[V2]
http://help.adobe.com/es_ES/as3/components/WS5b3ccc516d4fbf351e63e3d118a9c65b32-7f15.html

Puntos de referencia

-Ejemplo donde te enseña a convertir un video al formato .flv con el programa Adobe Media Encoder y a crear puntos de referencia incorporados.

[V3]
http://www.aulaclic.es/flash-cs4/t_21_1.htm

-Que son los cuepoints y como insertarlos en el video.

[V4]
http://desarrolloparaweb.blogspot.com/2010/02/anadir-interactividad-al-video-en-flash.html

-Ayuda de Adobe sobre los cuepoints

[V5]
http://help.adobe.com/es_ES/ActionScript/3.0_UsingComponentsAS3/WS5b3ccc516d4fbf351e63e3d118a9c65586-7feb.html

-Insertar cuepoints en el video para crear interactividad a través de botones

[V6]
http://www.upv.es/laboluz/2222/tecnica/botones_flash.htm

Exportar video con canal alfa

-Ajustes necesarios para codificar un .flv con canal alfa (transparencia)

[V7]
http://help.adobe.com/es_ES/premierepro/cs/using/WS2b635625315c8bcc1e63e3d1213c6c2363-8000.html#WS2b635625315c8bcc1e63e3d1213c6c2363-7ffe

Exportación de vídeo

[V8]
http://help.adobe.com/es_ES/mediaencoder/cs/using/WS2bacbdf8d487e582-30a3408e12f8ee21458-7ff2.html

CHROMA KEY

-Páginas sobre como realizar la técnica chroma key

[C1]
http://www.eradigital.com.ar/blog/archivos/croma_key.pdf

[C2]
http://www.loresdelsith.net/3po/rep/c_blue.htm

[C3]
http://www.maestrosdelweb.com/editorial/mejores-resultados-chroma-key/

[C4]
http://chanfainatv.com/?p=11131

-Iluminación a tres puntos

[C5]
http://www.maestrosdelweb.com/editorial/iluminacion-basica-video-web/

-Profundidad de campo

[C6]
http://130.206.170.120/pmmiRevisited/camara/pdfs/profundidad.pdf

-Chroma key con Adobe Premiere CS5

[C7]
http://www.google.es/search?client=safari&rls=en&q=chroma+key+con+adobe+premiere+pasos&ie=UTF-8&oe=UTF-8&redir_esc=&ei=DKtmTtTDA-HE4gTG8YHTCg

-Manual Premiere CS5

[C8]
http://help.adobe.com/es_ES/premierepro/cs/using/WS53d845b1b1b545acea-67d150e91260febee7a-8000.html

TEORÍA DE FLASH Y ACTIONSCRIP 3.0

[F1]
http://help.adobe.com/es_ES/ActionScript/3.0_ProgrammingAS3/WS5b3ccc516d4fbf351e63e3d118a9b90204-7fd9.html

Programación orientada a objetos.

[F2]
http://www.desarrolloweb.com/articulos/499.php

[F3]
http://luis.izqui.org/resources/ProgOrientadaObjetos.pdf

-Cursos completos de flash

[F4]
http://www.aulaclic.es/flash-cs4/index.htm

[F5]
http://www.estudiargratis.com.ar/webmaster/flash.htm

Clases en flash.

-Teoría sobre las clases en flash. Que son, como se usan y jerarquía.

[F6]
http://www.scribd.com/doc/1250904/Curso-ActionScript-30-en-Flash-CS3-AulaDirectiva

[F7]
http://www.cristalab.com/tips/introduccion-basica-a-actionscript-3-clases-tipos-de-datos-c29554l/

[F8]
http://www.esedeerre.com/ejemplo/9/94/utilizar-clases-con-actionscript-30-parte-i

[F9]
http://www.esedeerre.com/ejemplo/9/95/utilizar-clases-con-actionscript-30-parte-ii

Contenedores

-Teoría sobre los contenedores en flash, orden de jerarquía

[F10]
http://www.cristalab.com/tips/objetos-visibles-y-contenedores-en-actionscript-3-c46842l/

Tutoriales de Flash y ActionScript 3.0

-Página con muchos tutoriales interesantes y bien explicados

[F11]
http://www.esedeerre.com/portada/09/flash-ejemplos-y-tutoriales-de-actionscript-3.0-basico

-Recopilación de muchos tutoriales de flash y ActionScript 3.0

[F12]
http://www.forosdelweb.com/archive/index.php/f-16-p-36.html

-Tutoriales en flash, donde primero te explican la teoría.

[F13]
http://fermatflash.blogspot.com/

-Pagina completísima sobre casi todos los temas en flash y ActionScript 3.0, además tiene muchos artículos sobre flash.

[F14]
http://www.cristalab.com/tags/flash/

-Muchos tutoriales pero de cosas muy concretas, algunas parecen interesantes, todavía no lo he mirado bien.

[F15]
http://www.tutoriales-flash.com/

-Página con muchos tutoriales.

[F16]
http://www.solophotoshop.com/tutoriales/adobe-flash/

Manuales

[F17]
Programacin con ActionScript 3.0.

[F18]
Manual lenguaje ActionScript 3.0. componentes

[F19]
Utilización de componentes Adobe Actionscript 3.0

TUTORIALES ESPECÍFICOS PARA LAS PRUEBAS DE FLASH

[F20]
http://help.adobe.com/es_ES/ActionScript/3.0_ProgrammingAS3

Profundidad

-Teoría sobre la profundidad en flash.

[F21]
http://www.cristalab.com/tips/swapdepths-y-getnexthighestdepth-en-actionscript-3-c77580l/

-Tutoriales sobre el efecto de profundidad en flash

[F22]
http://www.sargentoweb.com/as3/?doc=16

[F23]
http://www.sargentoweb.com/as3/?doc=15

Botones

-Página donde te enseña a crear botones transparente para ampliar una imagen, pero se podría hacer cualquier cosa

[F24]
http://www.elwebmaster.com/editorial/taller-de-flash-botones-avanzados

-Tutorial que te enseña a crear botones con fade in y fade out. Interesante de ver.

[F25]
http://www.cristalab.com/tutoriales/boton-con-fade-in-y-fade-out-animado-c123l/

Interactividad con el ratón

-Tutoriales sobre como un objeto sigue la posición del puntero del ratón con diferentes efectos.

[F26]
http://www.sargentoweb.com/as3/?doc=3

[F27]
http://www.tutorials-expert.com/tutorial/23554/Follow-object-with-mouse-in-Actionscript-3.html

[F28]
http://www.esedeerre.com/ejemplo/9/119/seguir-el-puntero-del- raton-en-actionscript-30mi

-Ir a donde hago clic

[F29]
http://www.freeactionscript.com/tag/flash-game-click-to-move/

-Indicar posición del ratón cuando este se mueve

[F30]
http://www.flashvalley.com/fv_tutorials/mouse_control_and_actionscript/

-Tutorial sobre los tooltips (información cuando pasamos el ratón por encima de un objeto)

[F31]
http://www.sargentoweb.com/as3/?doc=40

-Seguir movimiento del ratón con clases

[F32]
http://asgamer.com/2009/character-movement-controls-follow-mouse

-Cambiar el cursor del ratón

[F33]
http://www.forosdelweb.com/f16/cambiar-puntero-mouse-s-3-0-a-513190/

-Coger y arrastrar con el ratón

[F34]
http://as3ideas.com/2010/02/06/drag-drop-en-as3-arrastrar-con-el-mouse/

[F35]
http://flash.astalaweb.net/Coger%20y%20arrastrar/1_Coger%20y%20arrastrar.asp

Dibujar

-Tutorial sobre como crear una hoja de dibujo en flash, tipo paint.

[F36]
http://www.sargentoweb.com/as3/?doc=11

Texto

-Tutorial sobre como crear áreas de textos, formularios

[F37]
http://www.sargentoweb.com/as3/?doc=30

-Como crear un campo de texto y modificar sus propiedades

[F38]
http://as3ideas.com/2010/03/22/textfield-en-as3-creacion-y-utilizacion-de-campos-de-texto-en-actionscript-3-parte-i/

[F39]
http://as3ideas.com/2010/04/20/darle-formato-a-un-texto-con-textformat-en-as3-defaulttextformat-en-as3/#more-791

-Como controlar los caracteres que podemos meter en un campo de texto

[F40]
http://www.webintenta.com/control-de-campos-de-texto-input-con-flash.html

Sonidos

-Ayuda de adobe sobre la utilización de sonidos en flash

[F41]
http://help.adobe.com/es_ES/flash/cs/using/WSd60f23110762d6b883b18f10cb1fe1af6-7ce8a.html

Interpolación de movimiento

-Introducir interpolación de movimiento en ActionScript 3.0

[F42]
http://active.tutsplus.com/tutorials/actionscript/an-introduction-to-tweening-with-actionscript-3-0/

[F43]
http://help.adobe.com/es_ES/ActionScript/3.0_ProgrammingAS3/WS13856C85-6B0D-447d-86DE-34DE1B70765C.html

Colisiones

[F44]
http://www.esedeerre.com/ejemplo/4/141/actionscript-30-colisiones-a-nivel-de-pixel

Filtros

[F45]
http://www.emanueleferonato.com/2009/04/28/understanding-as3-colormatrixfilter-class/

[F46]
http://help.adobe.com/es_ES/FlashPlatform/reference/actionscript/3/flash/filters/ColorMatrixFilter.html?filter_flash=cs5&filter_flashplayer=10.2&filter_air=2.6

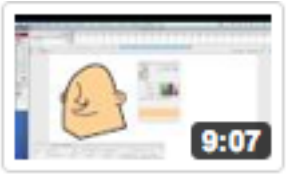
VIDEO TUTORIALES SOBRE ANIMACIONES EN PERSONAJES.

Los siguientes video tutoriales pertenecen al mismo autor. Los realiza para la página web flashfacilito.com

-Video tutorial que te da consejos para dibujar caras de personajes.

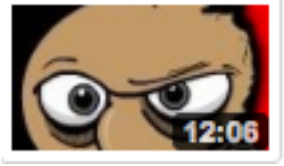
[85] http://www.youtube.com/watch?v=zJ9sM8GZJCQ&feature=related

[86] http://www.youtube.com/watch?v=km7qfX9L228&feature=related



-Video tutorial para animar las caras de los personajes

[87] http://www.youtube.com/watch?v=VeaAfHzrhvk&feature=channel



-Video tutorial que hace efecto Ed, Edd y Edy, temblar.

[88] http://www.youtube.com/watch?v=UKrvAJ8vGIE&feature=channel

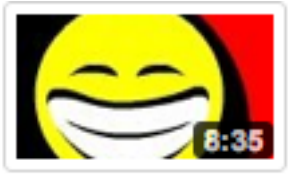
-Video tutorial para crear ojos.

[89] http://www.youtube.com/watch?v=3XsYM_GXpEI&feature=channel

-Video tutorial para crear emoticonos, gifs animados.

[90] http://www.youtube.com/watch?v=HCg26j1QEVQ&feature=channel

[91] http://www.youtube.com/watch?v=mWntfyVaIpU&feature=channel



-Video tutorial de efecto velocidad en los brazos.

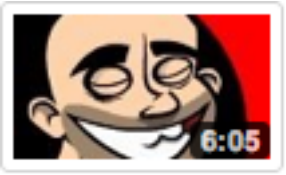
[92] http://www.youtube.com/watch?v=KoDIwVNTYz0&feature=channel

-Video tutorial para crear un movimiento lento de personaje que puede ser para cunado este parado.

[93] http://www.youtube.com/watch?v=MpN2nGSvfF8&feature=channel

-Video tutorial para crear el parpadeo de los ojos.

[94] http://www.youtube.com/watch?v=JsPrPE1WbyU&feature=related

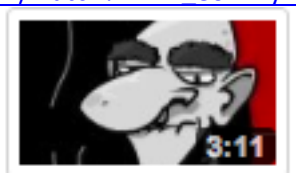


-Video tutorial para crear una sombra proyectada de un personaje.

[95] http://www.youtube.com/watch?v=VgwQVOU7HrM&feature=related

-Video tutorial para crear efecto antiguo en flash.

[96] http://www.youtube.com/watch?v=mk_eovEPyEg&feature=related



-Video tutorial para crear movimiento en personajes con esqueleto.

[97] http://www.youtube.com/watch?v=xS5kazveMrI&feature=related

-Video tutorial para crear efecto de velocidad, estela.

[98] http://www.youtube.com/watch?v=O47iBY18Kc4&feature=related

-Video tutorial interpolación de movimiento. Nueva forma de animar.

[99] http://www.youtube.com/watch?v=-6L6iu3RqGc&feature=related

-Video tutorial interpolación de forma.

[100] http://www.youtube.com/watch?v=O1Y2YU-S2fY&feature=related

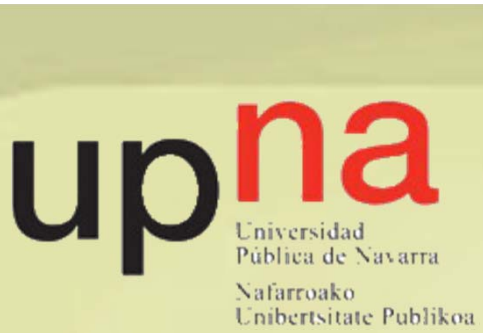
VIDEO TUTORIALES SOBRE EFECTOS EN FLASH SOBRE IMÁGENES REALES.

-Video tutorial sobre efecto de humo.

[101] http://www.youtube.com/watch?v=thbpdKJvOl4&feature=channel

-Video tutorial de gota realista en jarra de cerveza.

[102] http://www.youtube.com/watch?v=YgaipNES69k&feature=channel



ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Creación de una aventura gráfica integrando vídeo real

Proyecto final de carrera

Ingeniería técnica de telecomunicación, especialidad en sonido e imagen

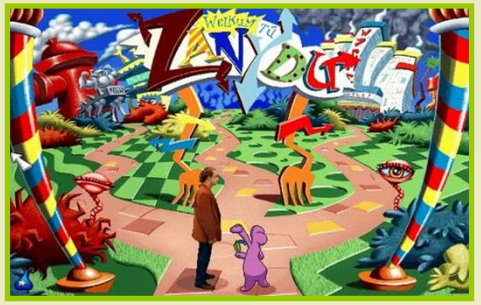
Tutor: Dr. Mikel Sagüés García

Irene Perea Castellanos

Descripción del proyecto

- Aventura gráfica en la que los escenarios y objetos están dibujados y el protagonista y los personajes son vídeos reales.

- En el mercado:



- Tecnologías utilizadas:

  -Adobe Flash CS5

  -ActionScript 3.0

  -Adobe Premiere CS5

  -Técnica Chroma key: Grabación de personaje sobre fondo verde y eliminación del fondo

ENSAYOS INICIALES

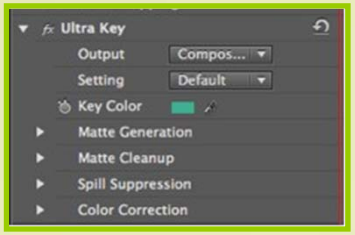
1. Integración de vídeo y uso de cuepoints

- Comprobar la posibilidad de incluir vídeo real con transparencia en Flash e interactuar con el.

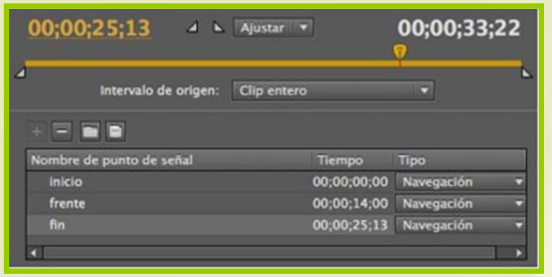
1. Integración de vídeo y uso de cuepoints

- Grabación sencilla

- Edición en Premiere con la herramienta Ultra key



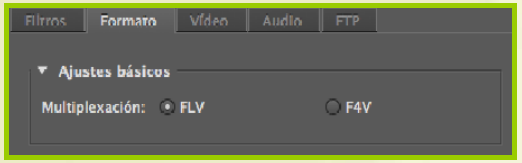
- Opciones de exportación



-Integración de cuepoints

-Parámetros de exportación

- FLV
- Codificar canal alpha



- Importar a Flash y colocar botones para interactuar con el vídeo

2. Creación de aplicaciones en Flash

- Importante pensar que se necesita programar para la aventura gráfica final

- En base a lo que se conocía y a la idea:

  -Elaboración de una lista de 16 aplicaciones

  - Mover objetos
  - Combinar
  - Interactividad del vídeo
  - …

2. Creación de aplicaciones en Flash

Ejemplo: Crear lugar para almacenar objetos

- Para cada aplicación se describió:
  - -Objetivo
  - -Objetos necesarios
  - -Estado inicial y final
  - -Diagrama de bloques

Estado inicial y final. Crear lugar para almacenar objetos.

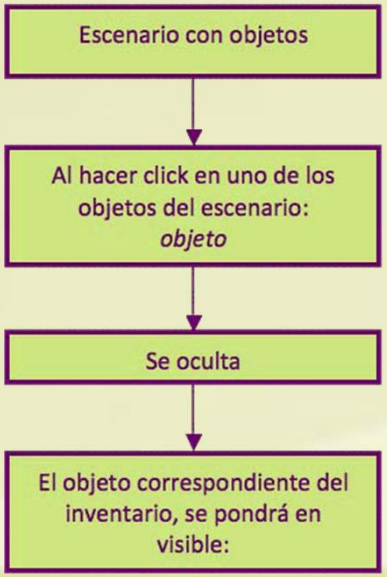


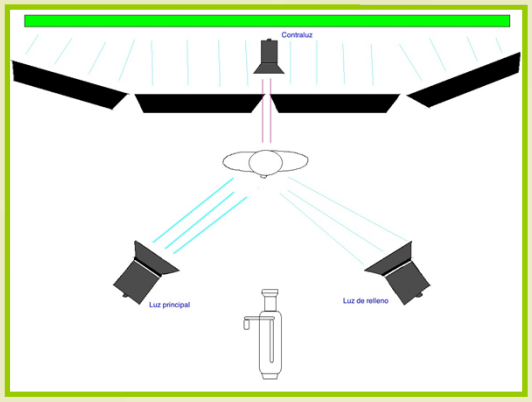
Diagrama. Crear lugar para almacenar objetos

3. Manejo de la cámara

- Conocer el funcionamiento de la cámara

  SONY PWM EX3

- Iluminación a tres puntos



- Grabaciones para estudiar los parámetros más importantes para saber controlarlos

Parámetros

- Balance de blancos
  - -Cartulina gris
  - -Cartulina blanca
- Ganancia
  - - -3dB
  - - 0dB
  - - 6dB
- Shutter
  - -Diferentes velocidades de shutter
    - Movimiento lentos
    - Movimientos rápidos
- Profundidad
  - -Manteniendo el fondo desenfocado
    - Mínima profundidad de campo
    - Mayor profundidad de campo
- Formatos
  - -HQ 1080/24P
  - -HQ720/24P
  - -HQ 1080/25P

4. Iluminación y edición

- Resultados de realizar un Chroma key con Iluminación a tres puntos
- Estudio de parámetros:
  - Herramienta ***Ultra Key*** de Premiere
  - Exportación
- Tres tipos de grabaciones:
  - Primer plano
  - Misma posición
  - Diferente posición

De frente en la misma posición. Resultados:

Para ver esta película, debe disponer de QuickTime™ y de un descompresor H.264.

5. Chroma key

- Decidir el encuadre del personaje que se iba a usar para las grabaciones finales

- Tres tipos de grabaciones:

  - Primer día
    - -Primer plano
    - -Plano americano
  - Segundo día
    - -Cuerpo entero

De perfil cuerpo entero.
Resultados:

Para ver esta película, debe disponer de QuickTime™ y de un descompresor H.264.

DISEÑO DE LA AVENTURA GRÁFICA

- Guión artístico
- Guión técnico

Guión artístico

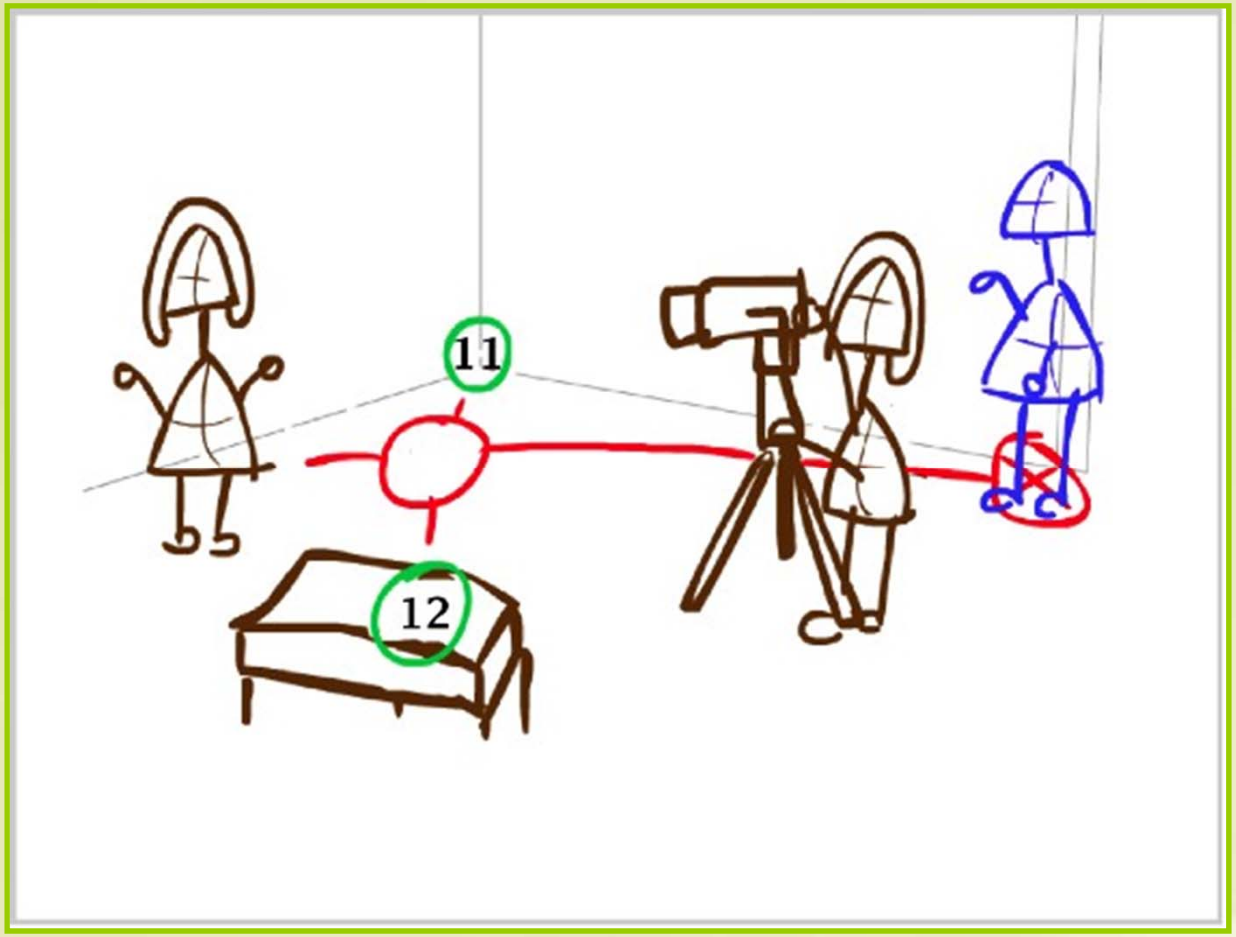
- Contar la historia

  -Invasión alienígena en la universidad

- Descripción breve

  -Escenarios
  -Objetos
  -Personajes
  -Movimientos del protagonista

Ejemplo escenario Chroma



Objetos:

-Gato chino (1
-Llave fija (1

Actores:

-Aloha bailan
-Irene graban

Gestos en movimien

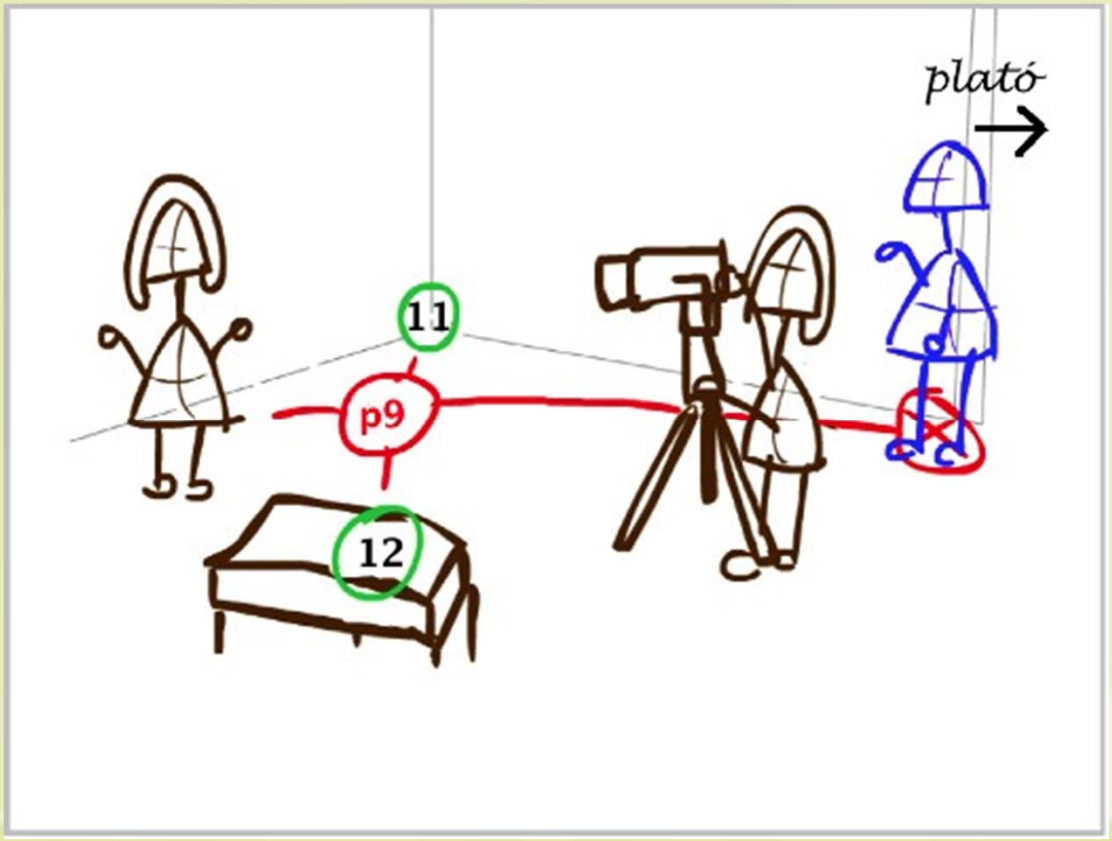
-De perfil derec
-De perfil izquier

Gestos quie

-De frente alargar la ma
-De espaldas agacha

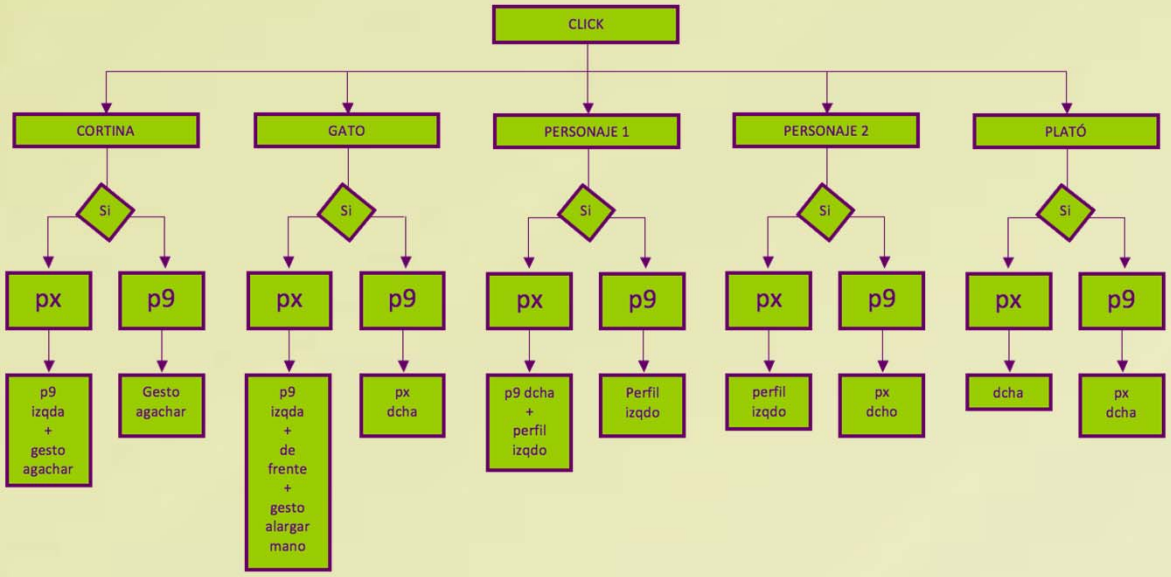
Guión técnico

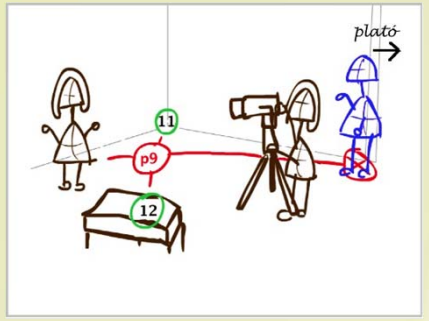
- Se explica lo mismo que en el artístico, de una manera más detallada



Diseño de la aventura gráfica. Guión técnico

➢ **Protagonista:** Describir las posiciones en las que realiza las acciones y los movimientos que realiza dependiendo de la posición en la que se encuentre.

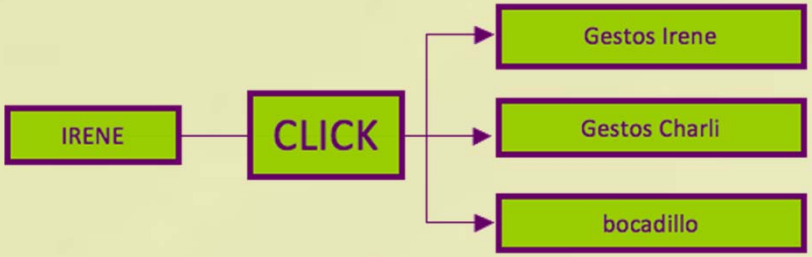




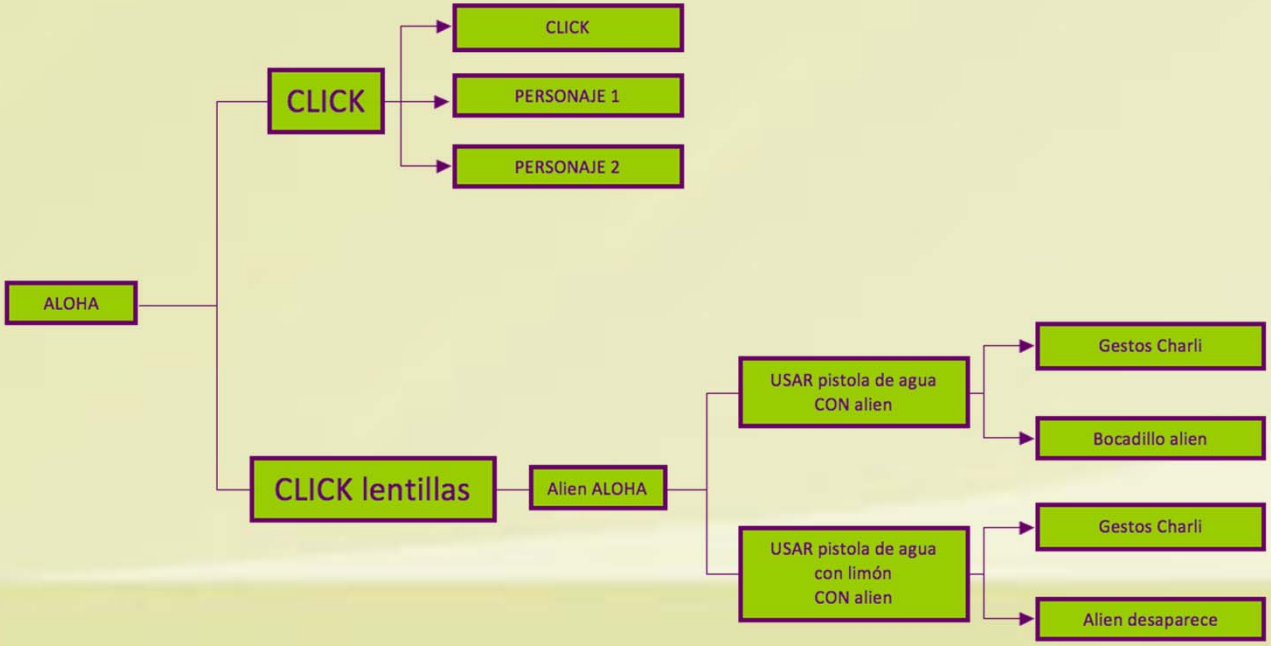
➢ **Objetos:** Describir las acciones que se pueden hacer con ellos. Coger, Usar o Combinar.

Diseño de la aventura gráfica. Guión técnico

- **Personajes:** Al hacer click sobre ellos, aparece un bocadillo de conversación los gestos de cada personaje cambian

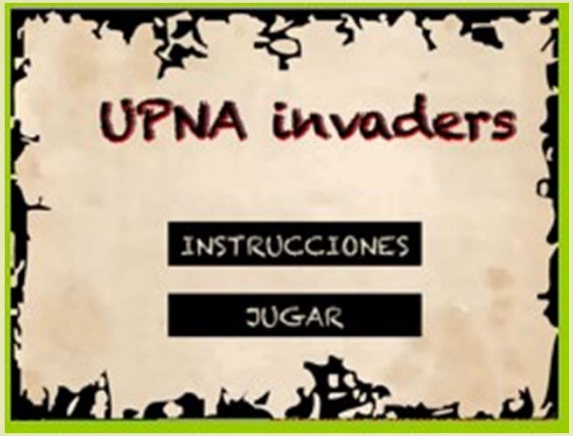


- **Aliens:** Visibles al hacer click en el objeto lentillas. Dependiendo de la pistola que se use con ellos, realizarán una acción u otra.



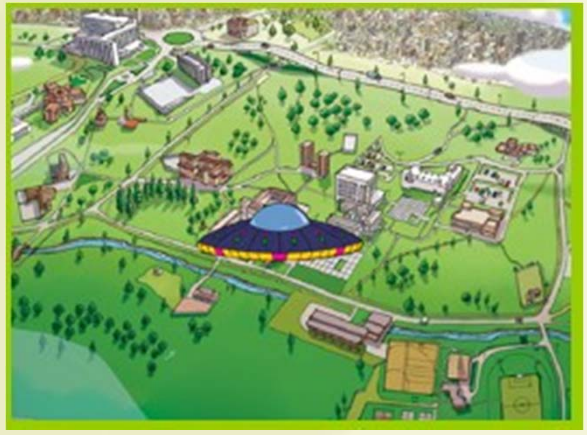
DESARROLLO DE LA AVENTURA GRÁFICA

- Aventura gráfica consta de 3 partes☐



Inicio

Juego



Animación final

Juego

- Dividido por escenarios
- Cada escenario en un fotograma
- Cada fotograma la programación
  - Inicialización de objetos
  - Llamadas a funciones de los objetos
  - Programación del los movimientos del personaje
  - Función **Objetos+Escenario()**

- Tipos de Objetos
  - Objetos del escenario
  - Objetos del inventario

Funciones

- Se podría decir que toda la aventura esta creada en base a 9 funciones diferentes
- La mayoría se basan en las aplicaciones que se crearon para las pruebas en Flash

Interactividad del personaje con los objetos

Para ver esta película, debe disponer de QuickTime™ y de un descompresor H.264.

Cambio de imagen del puntero y Texto del puntero

Para ver esta película, debe disponer de QuickTime™ y de un descompresor H.264.

Escribir y Dibujar

Para ver esta película, debe disponer de QuickTime™ y de un descompresor H.264.

Arrastrar objetos. Combinar/Usar

Para ver esta película, debe disponer de QuickTime™ y de un descompresor H.264.

Uso de filtros

Para ver esta película, debe disponer de QuickTime™ y de un descompresor H.264.

Carga de SWF

Para ver esta película, debe disponer de QuickTime™ y de un descompresor H.264.

CONCLUSIONES

- Generales

  -Importancia de buscar información, antes de comenzar a realizar el trabajo para poder estructurarlo bien.

  -Realización de las pruebas previas para conocer todos los aspectos que se tenían que tratar.

- Técnicas

  -Estudio previo del lenguaje AS3 al desarrollo de la aventura gráfica.

- Personales

  -Importancia de hacer algo que te gusta y que motiva.

  -Los temas que se trataban interesantes.

  -Aprendizaje.

¡GRACIAS POR VUESTRA ATENCIÓN!

(Aplausos)